Noname manuscript No.
(will be inserted by the editor)

An Iterative Adaptive *hp*-FEM Method for Non-Symmetric Elliptic Eigenvalue Problems

Pavel Solin · Stefano Giani

Abstract We present a novel adaptive higher-order finite element (*hp*-FEM) algorithm to solve non-symmetric elliptic eigenvalue problems. This is an extension of our prior work on symmetric elliptic eigenvalue problems. The method only needs to make one call to a generalized eigensolver on the coarse mesh, and then it employs Newton's or Picard's methods to resolve adaptively a selected eigenvalue-eigenvector pair. The fact that the method does not need to make repeated calls to a generalized eigensolver not only makes it very efficient, but it also eliminates problems that pose great complications to adaptive algorithms, such as eigenvalue reordering or returning arbitrary linear combinations of eigenvectors associated with the same eigenvalue. New theoretical and numerical results for the non-symmetric case are presented.

Keywords Partial differential equation · non-symmetric eigenvalue problem · iterative method · adaptive higher-order finite element method · *hp*-FEM

Mathematics Subject Classification (2000) 65N25 · 65N30 · 65N50

1 Introduction

This study presents a novel adaptive higher-order finite element (*hp*-FEM) algorithm for eigenvalue problems for non-symmetric elliptic partial differential equations (PDE). Eigenproblems are of considerable theoretical and practical interest in various areas of engineering and sciences. Classical applications include modal analysis in linear and nonlinear elasticity and the Schroedinger

P. Solin
Department of Mathematics and Statistics, University of Nevada, Reno, USA, and Institute of Thermomechanics, Prague, Czech Republic E-mail: solin.pavel@gmail.com

S. Giani
School of Mathematical Sciences, University of Nottingham, University Park, Nottingham, NG7 2RD, UK. E-mail: stefano.giani@nottingham.ac.uk

equation of quantum chemistry. But there are many other applications including, for example, automated multilevel sub-structuring methods for noise prediction in acoustics [14], analysis of photonic crystals [10,13] and plasmonic guides [5], stability analysis of fluid systems [6], and others.

The most common approach to solving eigenproblems is using eigensolvers. For larger problems it is practical to employ iterative eigensolvers such as ARPACK [15]. A characteristic common to all eigensolvers is that even if the user is interested in one particular eigenpair only, several additional eigenvalues and possibly eigenvectors need to be computed. These auxiliary eigenpairs are the byproduct of techniques such as deflation or orthogonalization that are used to filter out unwanted solutions.

Most eigensolvers are not designed to work with adaptive finite element methods (FEM), and their application on sequences of locally refined meshes can lead to substantial problems. In [19] we illustrated some of these problems and proposed a novel iterative method that alleviates them. The main idea was to adapt Picard's and Newton's methods to solve eigenvalue problems in order to minimize the number of unwanted eigenpairs exploiting the orthogonality between eigenvectors. In contrast to conventional adaptive methods that call an eigensolver after each mesh refinement, the iterative method is capable of following reliably a selected eigenpair on a sequence of adapted meshes. This is particularly useful with multiple eigenvalues when only a particular eigenfunction in the eigenspace is wanted.

In this paper we are going to extend the results from [19] to non-symmetric problems. The lack of symmetry brings substantial new complications that we need to overcome. For technical reasons, we have to assume that the model problem is a diagonalizable non-symmetric differential operator.

1.1 Model problem

We consider the eigenproblem

$$-\Delta u + b \cdot \nabla u + cu = \lambda u, \quad u = 0 \text{ on } \partial\Omega \tag{1}$$

where Ω is a bounded 2D domain with a Lipschitz-continuous boundary, $b \in \left[L^\infty(\Omega)\right]^2$ and $c \in L^\infty(\Omega)$ is non-negative.

1.2 Outline

The outline of the paper is as follows: In Section 2 we present motivation for the present study. In Section 3 we introduce notations and preliminaries. In Section 4 we present an algorithm based on Picard's method and demonstrate a need for orthogonalization. Sections 5 and 6 present Picard's and Newton's method with orthogonalization, respectively. Automatic *hp*-adaptivity is discussed in Section 7. Sections 8 and 9 present a reconstruction technology and

employ it to calculate reference solutions efficiently. Section 10 presents iterative methods with improved orthogonalization. Section 11 introduces a-priori convergence results. Numerical results are presented in Section 12.

2 Motivation

This work is motivated by the fact that non-symmetric eigenvalue problems in general have very different left and right eigenfunctions for the same eigenvalue. In Fig. 1 we show the left and right eigenfunctions corresponding to the smallest eigenvalue of problem (1) on the unit square with $b = (5, 5)$ and $c = 0$. As can be seen the two functions are different and in particular most of the energy of the modes is concentrated in different regions. It is obvious that these two eigenfunctions can not be approximated efficiently using the same mesh. Therefore, we use two independent sequences of refined meshes to approximate the left and right eigenfunctions.

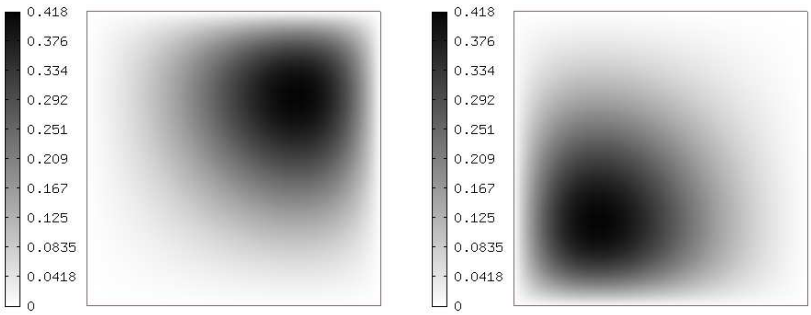

Fig. 1 Left and right eigenfunctions corresponding to the smallest eigenvalue of problem (1).

In Fig. 2 we show the *hp*-adapted meshes for the eigenfunctions from Fig. 1.

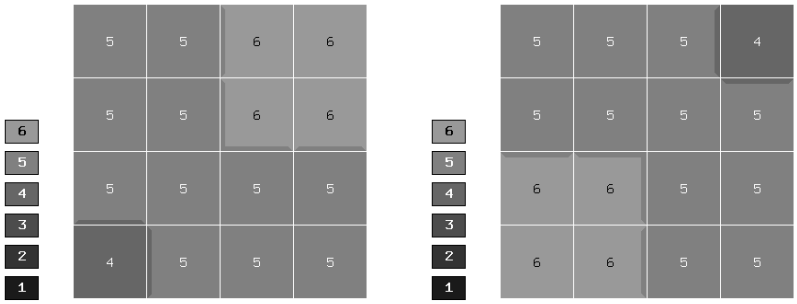

Fig. 2 Different *hp*-adapted meshes with polynomial degrees for the left and right eigenfunctions from Fig. 1.

3 Preliminaries

Throughout the paper, $L^2(\Omega)$ denotes the space of square-integrable real valued functions equipped with the standard norm

$$\|f\|_0 \;=\; \left(\int_\Omega |f|^2\right)^{\frac{1}{2}}. \tag{2}$$

The symbol $H^1(\Omega)$ denotes the space of functions in $L^2(\Omega)$ with square-integrable weak first partial derivatives. The H^1-norm is denoted by $\|f\|_1$.

Since problem (1) is non-symmetric, in general for the same eigenvalue the left and the right eigenfunctions are different. Therefore it is reasonable to compute for each eigenvalue both corresponding eigenfunctions, i.e., $(\lambda_j, u_j, u_j^*) \in \mathbb{R} \times E(\lambda_j) \times E^*(\lambda_j)$, where $E(\lambda_j)$ and $E^*(\lambda_j)$ are the left and right eigenspaces, respectively.

3.1 Weak formulation

The variational formulation of problem (1) reads: *Find the eigenvalue* $\lambda_j \in \mathbb{R}$ *and the eigenfunctions* $u_j, u_j^* \in H_0^1(\Omega)$ *such that*

$$\left.\begin{array}{lll} a(u_j, v) = \lambda_j\, b(u_j, v), & \text{for all} & v \in H_0^1(\Omega) \\ a(v, u_j^*) = \lambda_j\, b(v, u_j), & \text{for all} & v \in H_0^1(\Omega) \\ b(u_j, u_j^*) = 1 & & \end{array}\right\} \tag{3}$$

where

$$a(u, v) = \int_\Omega \nabla u(x) \cdot \nabla v(x) + b \cdot \nabla u(x) v(x) + c u(x) v(x), \tag{4}$$

and

$$b(u, v) = \int_\Omega u(x) v(x). \tag{5}$$

3.2 Finite element meshes

Since the left and the right eigenfunctions can be different, we use two sequences of meshes, one for each eigenfunction, and we apply adaptivity to each eigenfunction independently. To discretize (3), let $\mathcal{T}_n, n = 1, 2, \ldots$ denote a family of meshes on Ω. The meshes can be irregular with multiple levels of hanging nodes and they can combine possibly curvilinear triangular and quadrilateral elements [20]. These meshes may be obtained using automatic adaptivity. Similarly for the right eigenfunctions we use irregular mesh of the same kind. By $\mathcal{T}_n^*, n = 1, 2, \ldots$ we denote a family of meshes on Ω for the right eigenfunction.

In the rest of the paper we are going to denote with an asterisk '$*$' all quantities related to the right eigenfunction. By $h_{n,\tau}$ we denote the diameter of element τ and we define

$$h_n = \max_{\tau \in \mathcal{T}_n} \{h_{n,\tau}\}.$$

Similarly with $p_{n,\tau}$ we denote the order of polynomials of element τ, we define

$$p_n = \min_{\tau \in \mathcal{T}_n} \{p_{n,\tau}\}.$$

On any mesh $\mathcal{T}_n$ we denote by $V_n \subset H_0^1(\Omega)$ the finite dimensional space of continuous functions v such that on any element τ we have that $v|_\tau \in \mathcal{P}_{p_{n,\tau}}(\tau)$. Here either $\mathcal{P}_{p_{n,\tau}}(\tau)$ is the space of polynomials of total degree at most $p_{n,\tau}$ if τ is a triangular element, or $\mathcal{P}_{p_{n,\tau}}(\tau)$ is the space of polynomials of degree at most $p_{n,\tau}$ in each variable if τ is a quadrilateral element.

3.3 Discrete problem

The discrete version of (3) reads: *Find the eigenvalue* $\lambda_{j,n} \in \mathbb{R}$ *and the eigenfunctions* $u_{j,n} \in V_n$ *and* $u_{j,n}^* \in V_n^*$ *such that*

$$\left.\begin{array}{lll} a(u_{j,n}, v_n) = \lambda_{j,n}\, b(u_{j,n}, v_n), & \text{for all} & v_n \in V_n \\ a(v_n, u_{j,n}^*) = \lambda_{j,n}\, b(v_n, u_{j,n}^*), & \text{for all} & v_n \in V_n^* \\ b(u_{j,n}, u_{j,n}^*) = 1 & & \end{array}\right\} \tag{6}$$

3.4 Union mesh

We also make use of a third sequence of meshes $\mathcal{T}_n^U$ where for each n, $\mathcal{T}_n^U$ is the union of the meshes $\mathcal{T}_n$ and $\mathcal{T}_n^*$. Similarly we define the spaces V_n^U constructed on the meshes $\mathcal{T}_n^U$ which contain the spaces V_n and V_n^* for each n. The definition of the union mesh can be found, e.g., in [18]. In short, this is the smallest common mesh that is obtained by applying a sequence of standard refinement operations to the mesh $\mathcal{T}_n$ and that at the same time is obtained by another sequence of refinements applied to $\mathcal{T}_n^*$. The union mesh $\mathcal{T}_n^U$ is usually

finer than both $\mathcal{T}_n$ and $\mathcal{T}_n^*$. If $\mathcal{T}_n$ and $\mathcal{T}_n^*$ are the same, then $\mathcal{T}_n^U = \mathcal{T}_n$. If $\mathcal{T}_n$ can be obtained from $\mathcal{T}_n^*$ via a sequence of refinements, then $\mathcal{T}_n^U = \mathcal{T}_n$ and vice versa. For each element τ^U of the union mesh, there exists a unique element $\tau \in \mathcal{T}_n$ such that $\tau^U \subset \tau$, and a unique element $\tau^* \in \mathcal{T}_n^*$ such that $\tau^U \subset \tau^*$. The polynomial degree of τ^U is the maximum of the polynomial degrees of τ and τ^*.

For any function $v \in V_n$ we can define a trivial prolongation operator P such that $Pv \in V_n^U$ is the prolongation of v in V_n^U and similarly we can define the trivial prolongation operator P^* from V_n^* into V_n^U.

4 Picard's Method

Problem (6) can be reformulated in matrix form: *Find an eigenvalue* $\lambda \in \mathbb{R}$ *and an eigenvectors* $\mathbf{u} \in \mathbb{R}^N$*, where* N *is the dimension of* V_n *and* $\mathbf{u}^* \in \mathbb{R}^{N^*}$*, where* N^* *is the dimension of* V_n^**, such that*

$$\left.\begin{array}{ll} \mathbf{A}\mathbf{u} & = \lambda \mathbf{B}\mathbf{u} \ , \\ \mathbf{A}^*\mathbf{u}^* & = \lambda \mathbf{B}^*\mathbf{u}^* \ , \\ (P^*\mathbf{u}^*)^t \mathbf{B}^U P\mathbf{u} & = 1. \end{array}\right\} \tag{7}$$

Here the entries of the matrices $\mathbf{A}$ and $\mathbf{B}$ are

$$\mathbf{A}_{k,p} = a(\phi_p, \phi_k) \ , \quad \mathbf{B}_{k,p} = b(\phi_p, \phi_k) \ ,$$

where ϕ_i are the basis functions spanning V_n,

$$\mathbf{A}^*_{k,p} = a(\phi^*_k, \phi^*_p) \ , \quad \mathbf{B}^*_{k,p} = b(\phi^*_k, \phi^*_p) \ ,$$

where ϕ_i^* are the basis functions spanning V_n^* and

$$\mathbf{A}^U_{k,p} = a(\phi^U_k, \phi^U_p) \ , \quad \mathbf{B}^U_{k,p} = b(\phi^U_k, \phi^U_p) \ ,$$

where ϕ_i^U are the basis functions spanning V_n^U, the union finite element space.

The Picard method, presented in Algorithm 1, takes as arguments the matrices $\mathbf{A}$, $\mathbf{A}^*$, $\mathbf{A}^U$, $\mathbf{B}$, $\mathbf{B}^*$, $\mathbf{B}^U$, initial guesses $\tilde{u}$ and $\tilde{u}^*$ for the eigenfunctions, a relative tolerance Tol and an absolute tolerance AbsTol. The algorithm returns an approximate eigentriplet $(\lambda_{j,n}, u_{j,n}, u^*_{j,n})$. Because we use this iterative method on a sequence of adaptively refined meshes, we normally set as initial guesses the projections of the eigenfunctions of interest $u_{j,n-1}$ and $u^*_{j,n-1}$ on the refined meshes.

Algorithm 1 Picard's method

$(\lambda_{j,n}, u_{j,n}, u^*_{j,n}) = \text{Picard}(\mathbf{A}, \mathbf{A}^*, \mathbf{A}^U, \mathbf{B}, \mathbf{B}^*, \mathbf{B}^U, \tilde{u}, \tilde{u}^*, \text{Tol}, \text{AbsTol})$
$\mathbf{u}^1 = \tilde{u}$
$\mathbf{u}^{1,*} = \tilde{u}^*$
$\lambda^1 = \dfrac{(P^*\mathbf{u}^{1,*})^t \mathbf{A}^U P\mathbf{u}^1}{(P^*\mathbf{u}^{1,*})^t \mathbf{B}^U P\mathbf{u}^1}$
$m = 1$
repeat
  $\mathbf{u}^{m+1} = \mathbf{A}^{-1}\lambda^m \mathbf{B}\mathbf{u}^m$
  $\mathbf{u}^{m+1,*} = (\mathbf{A}^*)^{-1}\lambda^m \mathbf{B}^*\mathbf{u}^{m,*}$
  $\lambda^{m+1} = \dfrac{(P^*\mathbf{u}^{m+1,*})^t \mathbf{A}^U P\mathbf{u}^{m+1}}{(P^*\mathbf{u}^{m+1,*})^t \mathbf{B}^U P\mathbf{u}^{m+1}}$
  $m = m + 1$
until $\max\{\frac{\|\mathbf{u}^m - \mathbf{u}^{m-1}\|_1}{\|\mathbf{u}^{m-1}\|_1}, \frac{\|\mathbf{u}^{m,*} - \mathbf{u}^{m-1,*}\|_1}{\|\mathbf{u}^{m-1,*}\|_1}\} < \text{Tol}$ **or** $|\lambda^m - \lambda^{m-1}| < \text{AbsTol}$
$u_{j,n} = \mathbf{u}^m$
$u^*_{j,n} = \mathbf{u}^{m,*}$
$\lambda_{j,n} = \lambda^m$

The following theorem shows that Picard's method always converges to the smallest eigenvalue.

Theorem 1 *The Picard method in exact arithmetic converges into the eigenspace which is not orthogonal to the initial guesses* $\mathbf{u}^1, \mathbf{u}^{1,*}$ *and whose eigenvalue has minimum modulus.*

Proof Any left vector $\mathbf{u}^m$ can be expressed as

$$\mathbf{u}^m = \sum_{i=1}^{N} c_i^m \mathbf{u}_i \ ,$$

where c_i^m are real coefficients, N is the dimension of V_n and the vectors $\mathbf{u}_i \equiv u_{i,n}$ are the eigenvectors of the discrete problem, which form an orthonormal basis. Similarly any right vector $\mathbf{u}^{m,*}$ can be expressed as

$$\mathbf{u}^{m,*} = \sum_{i=1}^{N^*} c_i^{m,*} \mathbf{u}_i^* \ ,$$

where $c_i^{m,*}$ are real coefficients. Without loss in generality we can assume that λ_1 is the eigenvalue of minimum modulus and that c_1^1 and $c_1^{1,*}$ are different from 0.

In the case that λ_1 is simple we have from the definition of the problem:

$$\mathbf{u}^{m+1} = \mathbf{A}^{-1}\lambda^m \mathbf{B}\mathbf{u}^m = \left(\Pi_{j=1}^m \lambda^j\right)\left(\mathbf{A}^{-1}\mathbf{B}\right)^m \mathbf{u}^1 = \left(\Pi_{j=1}^m \lambda^j\right) \sum_{i=1}^{N} c_i^1 (\lambda_i)^{-m} \mathbf{u}_i \ ,$$

$$\mathbf{u}^{m+1,*} = (\mathbf{A}^*)^{-1}\lambda^m \mathbf{B}\mathbf{u}^{m,*} = \left(\Pi_{j=1}^m \lambda^j\right)\left((\mathbf{A}^*)^{-1}\mathbf{B}\right)^m \mathbf{u}^{1,*}$$

$$= \Big(\Pi_{j=1}^{m}\lambda^j\Big)\sum_{i=1}^{N^*} c_i^{1,*}(\lambda_i)^{-m}\mathbf{u}_i^* \ ,$$

where λ_i are the eigenvalues corresponding to $\mathbf{u}_i$, $\mathbf{u}_i^*$. Then

$$\mathbf{u}^{m+1} = \Big(\Pi_{j=1}^{m}\lambda^j\Big)(\lambda_1)^{-m}\Big(c_1^1\mathbf{u}_1 + \sum_{i=2}^{N} c_i^1\frac{(\lambda_1)^m}{(\lambda_i)^m}\mathbf{u}_i\Big) \ , \tag{8}$$

$$\mathbf{u}^{m+1,*} = \Big(\Pi_{j=1}^{m}\lambda^j\Big)(\lambda_1)^{-m}\Big(c_1^{1,*}\mathbf{u}_1^* + \sum_{i=2}^{N^*} c_i^{1,*}\frac{(\lambda_1)^m}{(\lambda_i)^m}\mathbf{u}_i^*\Big) \ , \tag{9}$$

where it is clear that, since $\lambda_1/\lambda_i < 1$, for $i \geq 2$, the directions of $\mathbf{u}^{m+1}$ and $\mathbf{u}^{m+1,*}$ tend toward the directions of $\mathbf{u}_1$, $\mathbf{u}_1^*$.

Then, exploiting the linearity of the prolongation operators P and P^* we have from (8) and (9):

$$P\mathbf{u}^{m+1} = \Big(\Pi_{j=1}^{m}\lambda^j\Big)(\lambda_1)^{-m}\Big(c_1^1 P\mathbf{u}_1 + \sum_{i=2}^{N} c_i^1\frac{(\lambda_1)^m}{(\lambda_i)^m}P\mathbf{u}_i\Big) \ ,$$

$$P^*\mathbf{u}^{m+1,*} = \Big(\Pi_{j=1}^{m}\lambda^j\Big)(\lambda_1)^{-m}\Big(c_1^{1,*}P^*\mathbf{u}_1^* + \sum_{i=2}^{N^*} c_i^{1,*}\frac{(\lambda_1)^m}{(\lambda_i)^m}P^*\mathbf{u}_i^*\Big) \ .$$

Furthermore, the Rayleigh quotient

$$\lambda^{m+1} = \frac{(P^*\mathbf{u}^{m+1,*})^t\mathbf{A}^U P\mathbf{u}^{m+1}}{(P^*\mathbf{u}^{m+1,*})^t\mathbf{B}^U P\mathbf{u}^{m+1}}$$

$$= \lambda_1\frac{(c_1^1c_1^{1,*})(P^*\mathbf{u}_1^*)^t\mathbf{B}^U P\mathbf{u}_1 + \displaystyle\sum_{i=2}^{N}(c_i^1c_i^{1,*})(P^*\mathbf{u}_i^*)^t\mathbf{B}^U P\mathbf{u}_i\left(\frac{\lambda_1}{\lambda_i}\right)^{2m-1}}{(c_1^1c_1^{1,*})(P^*\mathbf{u}_1^*)^t\mathbf{B}^U P\mathbf{u}_1 + \displaystyle\sum_{i=2}^{N}(c_i^1c_i^{1,*})(P^*\mathbf{u}_i^*)^t\mathbf{B}^U P\mathbf{u}_i\left(\frac{\lambda_1}{\lambda_i}\right)^{2m}} \ ,$$

converges to λ_1.

In the case that λ_1 has multiplicity R and that c_r^1, $c_r^{1,*}$, for some $1 \leq r \leq R$, are not zero, we similarly have that for all $i > R$:

$$P\mathbf{u}^{m+1} = \Big(\Pi_{j=1}^{m}\lambda^j\Big)(\lambda_1)^{-m}\Big(\sum_{r=1}^{R} c_r^1 P\mathbf{u}_r + \sum_{i=R+1}^{N} c_i^1\frac{(\lambda_1)^m}{(\lambda_i)^m}P\mathbf{u}_i\Big) \ ,$$

$$P^*\mathbf{u}^{m+1,*} = \Big(\Pi_{j=1}^{m}\lambda^j\Big)(\lambda_1)^{-m}\Big(\sum_{r=1}^{R} c_r^{1,*}P^*\mathbf{u}_r^* + \sum_{i=R+1}^{N^*} c_i^{1,*}\frac{(\lambda_1)^m}{(\lambda_i)^m}P^*\mathbf{u}_i^*\Big) \ ,$$

and then

$$\lambda^{m+1} = \frac{(P^*\mathbf{u}^{m+1,*})^t \mathbf{A}^U P\mathbf{u}^{m+1}}{(P^*\mathbf{u}^{m+1,*})^t \mathbf{B}^U P\mathbf{u}^{m+1}}$$

$$= \lambda_1 \frac{\displaystyle\sum_{r=1}^{R} (c_r^1 c_r^{1,*})(P^*\mathbf{u}_r^*)^t \mathbf{B}^U P\mathbf{u}_r + \sum_{i=R+1}^{N} (c_i^1 c_i^{1,*})(P^*\mathbf{u}_i^*)^t \mathbf{B}^U P\mathbf{u}_i \left(\frac{\lambda_1}{\lambda_i}\right)^{2m-1}}{\displaystyle\sum_{r=1}^{R} (c_r^1 c_r^{1,*})(P^*\mathbf{u}_r^*)^t \mathbf{B}^U P\mathbf{u}_r + \sum_{i=R+1}^{N} (c_i^1 c_i^{1,*})(P^*\mathbf{u}_i^*)^t \mathbf{B}^U P\mathbf{u}_i \left(\frac{\lambda_1}{\lambda_i}\right)^{2m}}$$

which converges again to λ_1.

Theorem 1 shows that even if the initial guesses $\mathbf{u}^1$ and $\mathbf{u}^{1,*}$ are very close to certain discrete eigenfunctions $u_{i,n}$ and $u_{i,n}^*$, for some i, the method can always converge to different eigenfunctions or linear combinations of eigenfunctions with corresponding eigenvalues smaller in modulus than $\lambda_{i,n}$. In real arithmetic, even if the initial guesses $\mathbf{u}^1$ and $\mathbf{u}^{1,*}$ are orthogonal to all eigenfunctions with index less than i, for some $m > 1$ the orthogonality could be perturbed, due to round-off errors, and the method can eventually converge anyway to different eigenfunctions or linear combinations of eigenfunctions with corresponding eigenvalues smaller in modulus than $\lambda_{i,n}$.

5 Picard's Method with Orthogonalization

In order to make Picard's method suitable to approximate efficiently any discrete eigentriplet, and not only the first one, we present Algorithm 2, which has an orthogonalization procedure in it. The orthogonalization procedure is specific for non-symmetric eigenvalue problems and it comes from the canonical form result for compact operators [22, Theorem 9.17].

The Picard method with orthogonalization takes as arguments the matrices $\mathbf{A}$, $\mathbf{A}^*$, $\mathbf{A}^U$, $\mathbf{B}$, $\mathbf{B}^*$, $\mathbf{B}^U$, initial guesses $\tilde{u}_{j,n-1}$ for $\tilde{u}_{j,n-1}^*$ for the eigenfunctions, which is the projection of the approximated eigenfunctions computed on the previous meshes, the tolerances AbsTol andTol and it also takes the $2(j-1)$ eigenfunctions $u_{1,n}, \ldots, u_{j-1,n}$ and $u_{1,n}^*, \ldots, u_{j-1,n}^*$. Then it returns the triplet $(\lambda_{j,n}, u_{j,n}, u_{j,n}^*)$ on the refined mesh.

Algorithm 2 Picard's method with orthogonalization

$(\lambda_{j,n}, u_{j,n}, u^*_{j,n}) = \text{PicardOrtho}(\mathbf{A}, \mathbf{A}^*, \mathbf{A}^U, \mathbf{B}, \mathbf{B}^*, \mathbf{B}^U, \tilde{u}_{j,n-1}, \tilde{u}^*_{j,n-1}, \text{Tol}, \text{AbsTol}, u_{1,n}, \ldots, u_{j-1,n}, u^*_{1,n}, \ldots, u^*_{j-1,n})$
$\mathbf{u}^1 = \tilde{u}_{j,n-1}$
$\mathbf{u}^{1,*} = \tilde{u}^*_{j,n-1}$
$\lambda^1 = \dfrac{(P^*\mathbf{u}^{1,*})^t \mathbf{A}^U P\mathbf{u}^1}{(P^*\mathbf{u}^{1,*})^t \mathbf{B}^U P\mathbf{u}^1}$
$m = 1$
repeat
  $\mathbf{u}^{m+1} = \mathbf{A}^{-1}\lambda^m \mathbf{B}\mathbf{u}^m$
  $\mathbf{u}^{m+1,*} = (\mathbf{A}^*)^{-1}\lambda^m \mathbf{B}^*\mathbf{u}^{m,*}$
  for $i = 1$ to $j-1$ **do**
    $\mathbf{u}^{m+1} = \mathbf{u}^{m+1} - ((P^*u^*_{i,n})^t \mathbf{B}^U P\mathbf{u}^{m+1})u_{i,n}$ {Orthogonalization}
    $\mathbf{u}^{m+1,*} = \mathbf{u}^{m+1,*} - (P^*u^t_{i,n} \mathbf{B}^U P\mathbf{u}^{m+1,*})u^*_{i,n}$ {Orthogonalization}
  end for
  $\mathbf{u}^{m+1} = \dfrac{\mathbf{u}^{m+1}}{|(P^*\mathbf{u}^{m+1,*})^t \mathbf{B}^U P\mathbf{u}^{m+1}|^{1/2}}$ {Normalization}
  $\mathbf{u}^{m+1,*} = \dfrac{\mathbf{u}^{m+1,*}}{|(P^*\mathbf{u}^{m+1,*})^t \mathbf{B}^U P\mathbf{u}^{m+1}|^{1/2}}$ {Normalization}
  $\lambda^{m+1} = \dfrac{(P^*\mathbf{u}^{m+1,*})^t \mathbf{A}^U P\mathbf{u}^{m+1}}{(P^*\mathbf{u}^{m+1,*})^t \mathbf{B}^U P\mathbf{u}^{m+1}}$
  $m = m + 1$
until $\max\{\frac{\|\mathbf{u}^m - \mathbf{u}^{m-1}\|_1}{\|\mathbf{u}^{m-1}\|_1}, \frac{\|\mathbf{u}^{m,*} - \mathbf{u}^{m-1,*}\|_1}{\|\mathbf{u}^{m-1,*}\|_1}\} < \text{Tol}$ **or** $|\lambda^m - \lambda^{m-1}| < \text{AbsTol}$
$u_{j,n} = \mathbf{u}^m$
$u^*_{j,n} = \mathbf{u}^{m,*}$
$\lambda_{j,n} = \lambda^m$

As can be seen in Algorithm 2, the orthogonalization is done in each iteration. This is necessary in real arithmetic to guarantee that $\mathbf{u}^m$ and $\mathbf{u}^{m,*}$ are orthogonal to all eigenfunctions $u_{1,n}, \ldots, u_{j-1,n}$ and $u^*_{1,n}, \ldots, u^*_{j-1,n}$, for all m. However in exact arithmetic it would be enough to orthogonalize only $\mathbf{u}^1$ and $\mathbf{u}^{1,*}$. The normalization step is necessary in all iterations because, due to the orthogonalization procedure, this version of Picard's method does not conserve the norm of the vectors and possible underflows or overflows could happen with no normalization.

Theorem 2 *Algorithm 2 never converges to an eigenvalue of index smaller than j.*

Proof The proof comes straightforwardly from the arguments used to prove Theorem 1. The fact that $\mathbf{u}^m$ is orthogonal to all eigenfunctions $\mathbf{u}_1, \ldots, \mathbf{u}_{j-1}$, implies that the coefficients c_i^m, with $m = 1, \ldots, j-1$, are zeros. Similarly $\mathbf{u}^{m,*}$ is orthogonal to all eigenfunctions $\mathbf{u}^*_1, \ldots, \mathbf{u}^*_{j-1}$, which implies that the coefficients $c_i^{m,*}$, with $m = 1, \ldots, j-1$, are zeros. Then, the Rayleigh quotient converges to λ_j by the same arguments used before.

6 Newton's Method with Orthogonalization

The second iterative method that we are going to propose is based on Newton's method applied to eigenvalue problems, see Algorithm 3. Denoting $\tilde{x} = (x, \lambda)$, we have that problem (3) for the left eigenfunction can be rewritten in the form

$$0 = f(\tilde{x}) = \begin{pmatrix} Ax & - \lambda\, Bx \\ x^T Bx & - 1 \end{pmatrix}.$$

Then denoting by $\tilde{h} = (h, \delta)^t$ the increment, we have that the truncated Taylor series of the problem is

$$f(\tilde{x} + \tilde{h}) \approx f(\tilde{x}) + J_f(\tilde{x}) \cdot \tilde{h}, \tag{10}$$

where the Jacobian matrix is defined as

$$J_f(\tilde{x}) = \begin{pmatrix} A - B\lambda & -Bx \\ 2Bx^T & 0 \end{pmatrix}.$$

If $\tilde{x} + \tilde{h}$ is a solution of (3) for the left eigenfunction, we have from (10) that

$$J_f(\tilde{x}) \cdot \tilde{h} = -f(\tilde{x}),$$

which defines the linear problem of Newton's method. Similarly, denoting by $\tilde{x}^* = (x^*, \lambda)$, we have that problem (3) for the right eigenfunction can be rewritten in the form

$$0 = f^*(\tilde{x}^*) = \begin{pmatrix} A^* x^* & - \lambda\, B^* x^* \\ (x^*)^T B^* x^* & - 1 \end{pmatrix}.$$

Finally, denoting by $\tilde{h}^* = (h^*, \delta)^t$ the increment, we have that the truncated Taylor series of the problem is

$$f^*(\tilde{x}^* + \tilde{h}^*) \approx f^*(\tilde{x}^*) + J_f^*(\tilde{x}^*) \cdot \tilde{h}^*, \tag{11}$$

where the Jacobian matrix is defined as

$$J_f^*(\tilde{x}^*) = \begin{pmatrix} A^* - B^*\lambda & -B^* x^* \\ 2B^*(x^*)^T & 0 \end{pmatrix}.$$

If $\tilde{x}^* + \tilde{h}^*$ is a solution of (3) for the right eigenfunction, we have from (11) that

$$J_f^*(\tilde{x}^*) \cdot \tilde{h}^* = -f^*(\tilde{x}^*),$$

which defines the linear problem of Newton's method.

In order to make the method suitable for all eigenpairs, we are going to write a version of Newton's method that uses an orthogonalization procedure, similarly to what we have already done for Picard's method.

Theorem 3 *Algorithm 4 always converges to an eigenvalue greater or equal to λ_j.*

Algorithm 3 Newton's method

$(\lambda_{j,n}, u_{j,n}, u^*_{j,n}) = \text{Newton}(\mathbf{A}, \mathbf{A}^*, \mathbf{A}^U, \mathbf{B}, \mathbf{B}^*, \mathbf{B}^U, \tilde{u}, \tilde{u}^*, \text{Tol}, \text{AbsTol})$
$\mathbf{u}^1 = \tilde{u}$
$\mathbf{u}^{1,*} = \tilde{u}^*$
$\lambda^1 = \dfrac{(P^*\mathbf{u}^{1,*})^t \mathbf{A}^U P\mathbf{u}^1}{(P^*\mathbf{u}^{1,*})^t \mathbf{B}^U P\mathbf{u}^1}$
$m = 1$
repeat
  Solve $J_f(\mathbf{u}^m, \lambda^m) \cdot \tilde{h} = -f(\mathbf{u}^m, \lambda^m)$
  $\mathbf{u}^{m+1} = \mathbf{u}^m + h$
  $\lambda^{m+1} = \lambda^m + \delta$
  Solve $J^*_f(\mathbf{u}^{m,*}, \lambda^{m+1}) \cdot \tilde{h}^* = -f^*(\mathbf{u}^{m,*}, \lambda^{m+1})$
  $\lambda^{m+1} = \lambda^{m+1} + \delta$
  $\mathbf{u}^{m+1,*} = \mathbf{u}^{m,*} + h^*$
  $m = m + 1$
until $\max\{\frac{\|\mathbf{u}^m - \mathbf{u}^{m-1}\|_1}{\|\mathbf{u}^{m-1}\|_1}, \frac{\|\mathbf{u}^{m,*} - \mathbf{u}^{m-1,*}\|_1}{\|\mathbf{u}^{m-1,*}\|_1}\} < \text{Tol}$ **or** $|\lambda^m - \lambda^{m-1}| < \text{AbsTol}$
$u_{j,n} = \mathbf{u}^m$
$u^*_{j,n} = \mathbf{u}^{m,*}$
$\lambda_{j,n} = \lambda^m$

Algorithm 4 Newton's method with orthogonalization

$(\lambda_{j,n}, u_{j,n}, u^*_{j,n}) = \text{NewtonOrtho}(\mathbf{A}, \mathbf{A}^*, \mathbf{A}^U, \mathbf{B}, \mathbf{B}^*, \mathbf{B}^U, \tilde{u}_{j,n-1}, \tilde{u}^*_{j,n-1}, \text{Tol}, \text{AbsTol},$
  $u_{1,n}, \ldots, u_{j-1,n}, u^*_{1,n}, \ldots, u^*_{j-1,n})$
$\mathbf{u}^1 = \tilde{u}_{j,n-1}$
$\mathbf{u}^{1,*} = \tilde{u}^*_{j,n-1}$
$\lambda^1 = \dfrac{(P^*\mathbf{u}^{1,*})^t \mathbf{A}^U P\mathbf{u}^1}{(P^*\mathbf{u}^{1,*})^t \mathbf{B}^U P\mathbf{u}^1}$
$m = 1$
repeat
  Solve $J_f(\mathbf{u}^m, \lambda^m) \cdot \tilde{h} = -f(\mathbf{u}^m, \lambda^m)$
  $\mathbf{u}^{m+1} = \mathbf{u}^m + h$
  $\lambda^{m+1} = \lambda^m + \delta$
  Solve $J^*_f(\mathbf{u}^{m,*}, \lambda^{m+1}) \cdot \tilde{h}^* = -f^*(\mathbf{u}^{m,*}, \lambda^{m+1})$
  $\lambda^{m+1} = \lambda^{m+1} + \delta$
  $\mathbf{u}^{m+1,*} = \mathbf{u}^{m,*} + h^*$
  for $i = 1$ to $j - 1$ **do**
    $\mathbf{u}^{m+1} = \mathbf{u}^{m+1} - ((u^*_{i,n})^t \mathbf{B}\mathbf{u}^{m+1}) u_{i,n}$ {Orthogonalization}
    $\mathbf{u}^{m+1,*} = \mathbf{u}^{m+1,*} - (u^t_{i,n} \mathbf{B}\mathbf{u}^{m+1,*}) u^*_{i,n}$ {Orthogonalization}
  end for
  $\mathbf{u}^{m+1} = \dfrac{\mathbf{u}^{m+1}}{|(P^*\mathbf{u}^{m+1,*})^t \mathbf{B}^U P\mathbf{u}^{m+1}|^{1/2}}$ {Normalization}
  $\mathbf{u}^{m+1,*} = \dfrac{\mathbf{u}^{m+1,*}}{|(P^*\mathbf{u}^{m+1,*})^t \mathbf{B}^U P\mathbf{u}^{m+1}|^{1/2}}$ {Normalization}
  $\lambda^{m+1} = \dfrac{(P^*\mathbf{u}^{m+1,*})^t \mathbf{A}^U P\mathbf{u}^{m+1}}{(P^*\mathbf{u}^{m+1,*})^t \mathbf{B}^U P\mathbf{u}^{m+1}}$
  $m = m + 1$
until $\max\{\frac{\|\mathbf{u}^m - \mathbf{u}^{m-1}\|_1}{\|\mathbf{u}^{m-1}\|_1}, \frac{\|\mathbf{u}^{m,*} - \mathbf{u}^{m-1,*}\|_1}{\|\mathbf{u}^{m-1,*}\|_1}\} < \text{Tol}$ **or** $|\lambda^m - \lambda^{m-1}| < \text{AbsTol}$
$u_{j,n} = \mathbf{u}^m$
$u^*_{j,n} = \mathbf{u}^{m,*}$
$\lambda_{j,n} = \lambda^m$

Proof This result is a direct consequence of the orthogonalization step in Algorithm 4. We are using again the fact that any right vector $\mathbf{u}^{m+1}$ can be expressed as

$$\mathbf{u}^{m+1} = \sum_{i=1}^{N} c_i^{m+1} \mathbf{u}_i,$$

where c_i^{m+1} are real coefficients and the vectors $\mathbf{u}_i \equiv u_{i,n}$ are the eigenvectors of the discrete problem, which are sorted accordingly the magnitude of the corresponding eigenvalues λ_i. Similarly any left vector $\mathbf{u}^{m+1,*}$ can be expressed as

$$\mathbf{u}^{m+1,*} = \sum_{i=1}^{N^*} c_i^{m+1,*} \mathbf{u}_i^*,$$

where c_i^{m+1} are real coefficients and the vectors $\mathbf{u}_i^* \equiv u_{i,n}^*$ are the eigenvectors of the discrete problem, which are sorted accordingly the magnitude of the corresponding eigenvalues λ_i In particular, when $\mathbf{u}^{m+1} = \mathbf{u}^m + h$, $\mathbf{u}^{m+1,*} = \mathbf{u}^{m,*} + h^*$, we have that, after the application of the orthogonalization step, the resulting vectors are

$$\hat{\mathbf{u}}^{m+1} = \sum_{i=j}^{N} c_i^{m+1} \mathbf{u}_i \ ,$$

and

$$\hat{\mathbf{u}}^{m+1,*} = \sum_{i=j}^{N^*} c_i^{m+1,*} \mathbf{u}_i^* \ .$$

Then, it is straightforward to see that the Rayleigh quotient

$$\lambda^{m+1} = \frac{(P^* \hat{\mathbf{u}}^{m+1,*})^t \mathbf{A}^U P \hat{\mathbf{u}}^{m+1}}{(P^* \hat{\mathbf{u}}^{m+1,*})^t \mathbf{B}^U P \hat{\mathbf{u}}^{m+1}} \geq \lambda_j.$$

7 Automatic *hp*-Adaptivity

With the Picard and Newton methods in hand, we can now proceed to automatic *hp*-adaptivity. This part of the paper is not new but we need to present it to make the paper self-contained. We use an algorithm from [20] that is an analogy to embedded higher-order ODE methods: In each adaptivity step we construct an approximation pair with different orders of accuracy and use their difference as an a-posteriori error estimator.

Algorithm 5 Automatic *hp*-adaptivity

Let $\mathcal{T}_0^c$ and $\mathcal{T}_0^{c,*}$ be the initial coarse meshes. We construct the initial fine meshes $\mathcal{T}_0^f$ and $\mathcal{T}_0^{f,*}$ by refining all elements in space and moreover increasing their polynomial degrees by one. A generalized eigensolver is called one time only, to obtain a solution triplet $(\lambda_0^c, u_0^c, u_0^{c,*})$ on the initial coarse meshes $\mathcal{T}_0^c$ and $\mathcal{T}_0^{c,*}$.
Set $k = 0$
repeat
Project the approximation u_k^c to the mesh $\mathcal{T}_k^f$ and the approximation $u_k^{c,*}$ to the mesh $\mathcal{T}_k^{f,*}$. The projection is denoted by $P_k^f u_k^c$ and by $P_k^{f,*} u_k^{c,*}$. Since the finite element spaces on meshes $\mathcal{T}_k^c$ and $\mathcal{T}_k^f$ are embedded as well as $\mathcal{T}_k^{c,*}$ and $\mathcal{T}_k^{f,*}$, there is no projection error.
Calculate an initial guess $\tilde{\lambda}_k^f$ for the eigenvalue using the relation

$$\tilde{\lambda}_k^f = \frac{(P^* P_k^{f,*} u_k^{c,*})^T \mathbf{A}_k^{U,f} P P_k^f u_k^c}{(P^* P_k^{f,*} u_k^{c,*})^T \mathbf{B}_k^{U,f} P P_k^f u_k^c},$$

where $\mathbf{A}_k^{U,f}$ and $\mathbf{B}_k^{U,f}$ are the stiffness and mass matrices on the union mesh constructed from $\mathcal{T}_k^f$ and $\mathcal{T}_k^{f,*}$.
The triplet $(\tilde{\lambda}_k^f, P_k^f u_k^c, P_k^{f,*} u_k^{c,*})$ is **not a solution** to the generalized eigenproblems on the mesh $\mathcal{T}_k^f$ and $\mathcal{T}_k^{f,*}$, but it is used as an initial guess.
Apply Picard's or Newton's method as described in Sections 5 and 6, to obtain a solution triplet $(\lambda_k^f, u_k^f, u_k^{f,*})$ on the meshes $\mathcal{T}_k^f$ and $\mathcal{T}_k^{f,*}$.
Project the approximations u_k^f and $u_k^{f,*}$ back to the coarse meshes $\mathcal{T}_k^c$ and $\mathcal{T}_k^{c,*}$ to obtain $P_k^c u_k^f$ and $P_k^{c,*} u_k^{f,*}$.
Calculate the a-posteriori error estimates e_k^c and $e_k^{c,*}$,

$$e_k^c = u_k^f - P_k^c u_k^f, \quad e_k^{c,*} = u_k^{f,*} - P_k^{c,*} u_k^{f,*}.$$

Note: e_k^c and $e_k^{c,*}$ are functions, not numbers.
Use e_k^c and $e_k^{c,*}$ to guide one step of automatic *hp*-adaptivity [20] that yields the new coarse meshes $\mathcal{T}_{k+1}^f$ and $\mathcal{T}_{k+1}^{f,*}$.
Update $k = k + 1$
until The H^1-norms of e_{k-1}^c and $e_{k-1}^{c,*}$ are sufficiently small.

8 Reconstruction Technology

It is well known that the discretization process perturbs the spectrum, in particular a left eigenspace $E(\lambda_j)$ of multiple eigenvalue λ_j can be split in more than one discrete eigenspaces $E(\lambda_{j,n}), E(\lambda_{j+1,n}), \ldots, E(\lambda_{j+m,n})$ with correspondent discrete eigenvalues $\lambda_{j,n}, \lambda_{j+1,n}, \ldots, \lambda_{j+m,n}$ forming a small cluster for sufficiently rich finite element spaces, also under the same assumption we have that

$$\dim E(\lambda_j) = \sum_{i=0}^{m} \dim E(\lambda_{j+i,n}).$$

The same could happen to any right eigenspace $E^*(\lambda_j)$. This phenomenon is already well documented in literature, see [21,3,12].

Different finite element spaces can split the same multiple eigenspace in different ways, this also happens with adaptively refined meshes. It is not

rare that the same multiple eigenspace is split differently on the coarse and on the refined mesh. Since a different split corresponds to different discrete eigenfunctions, then it is not always possible to find for the same eigenvalue on the refined mesh an eigenfunction similar to the one on the coarse mesh. It is crucial for the adaptive algorithm to prevent this behavior.

Therefore we propose a way to always construct on the refined mesh an approximation of the same eigenfunction as on the coarse mesh. The idea is based on the fact that for sufficiently rich finite element spaces, the spaces

$$M_n(\lambda_j) = \text{span}\{E(\lambda_{j,n}), E(\lambda_{j+1,n}), \ldots, E(\lambda_{j+m,n})\}$$

and

$$M_n^*(\lambda_j) = \text{span}\{E^*(\lambda_{j,n}), E^*(\lambda_{j+1,n}), \ldots, E^*(\lambda_{j+m,n})\}$$

are approximations of the spaces $E(\lambda_j)$ and $E^*(\lambda_j)$, see [3]. So for any couple $(U_{n-1}, U_{n-1}^*) \in M_{n-1}(\lambda_j) \times M_{n-1}^*(\lambda_j)$, we propose the couple $(U_n, U_n^*) \in M_n(\lambda_j) \times M_n^*(\lambda_j)$ such as

$$U_n = \sum_{i=1}^{R} c_i \; u_{i,n}, \tag{12}$$

where $u_{1,n}, u_{2,n}, \ldots, u_{R,n}$, with $R = \dim E(\lambda_j) = \dim E^*(\lambda_j)$, are eigenfunctions of the discrete problem forming an orthonormal basis for M_n. Similarly,

$$U_n^* = \sum_{i=1}^{R} c_i^* \; u_{i,n}^*, \tag{13}$$

where $u_{1,n}^*, u_{2,n}^*, \ldots, u_{R,n}^*$, are eigenfunctions of the discrete problem forming an orthonormal basis for M_n^*. The coefficients c_i and c_i^* satisfy

$$\sum_{i=1}^{R} c_i c_i^* = 1, \tag{14}$$

and by construction we have that $b(u_{i,n}, u_{i,n}^*) = 1$ for any $i = 1, \ldots, R$.

From the definition of problem (3) we have that the reconstructed eigenvalue is defined as

$$\Lambda_n = \frac{a(U_n, U_n^*)}{b(U_n, U_n^*)}.$$

The triplet (Λ_n, U_n, U_n^*) is not a discrete eigentriplet of problem (6) in general. In Section 11 we prove that (Λ_n, U_n, U_n^*) converges a priori at the same rate as any other discrete eigentriplet of (3) to the continuous eigentriplet.

9 Computing Reference Solution via Reconstruction

In this section we present two algorithms to compute approximations of eigenpairs. Each algorithm is based on a different method to compute the discrete spectrum, but both of them use the reconstruction technology to keep track of the eigenfunction of interest.

In all algorithms we are going to use on the initial mesh an iterative eigensolver with calling interface $\{(\lambda_{j,n}, u_{j,n}, u^*_{j,n})_{j=1}^{i}\} = \text{Eigensolver}(\mathbf{A}, \mathbf{A}^*, \mathbf{B}, \mathbf{B}^*, i, \text{Tol}, \text{MaxIter})$, that computes the set of discrete triplets $\{(\lambda_{j,n}, u_{j,n}, u^*_{j,n})\}_{j=1}^{i}$ and where $\mathbf{A}$ and $\mathbf{A}^*$ are the stiffness matrices of the problems, $\mathbf{B}$ and $\mathbf{B}^*$ are the mass matrices of the problems, i is the number of eigenpairs to compute, Tol is the requested tolerance for the eigenpairs and MaxIter is the maximum number of iterations.

All algorithm we describe below are based on the reconstruction technology which is guided by two parameters: DTE and FIE. The parameter DTE should be equal to the multiplicity of the continuous eigenvalue λ that the user wants to approximate. All algorithms work also when DTE is an upper bound of the multiplicity of λ, so in practice the multiplicity of the target eigenvalue is not necessary to be known exactly. The parameter FIE should be equal to the index i of the first discrete eigenvalue on the initial mesh $\lambda_{i,0}$ that approximates λ. The reconstruction technology is described in Algorithm (6).

Algorithm 6 Reconstruction algorithm

$(\Lambda_n, U_n, U^*_n) = \text{Reconstruction}(\{(\lambda_{j,n}, u_{j,n}, u^*_{j,n})\}_{j=\text{FIE}}^{\text{FIE+DTE}}, (\Lambda_{n-1}, U_{n-1}, U^*_{n-1}))$

Compute $U_n = \displaystyle\sum_{i=\text{FIE}}^{\text{FIE+DTE}} b(U_{n-1}, u^*_{i,n}) u_{i,n}$

Compute $U^*_n = \displaystyle\sum_{i=\text{FIE}}^{\text{FIE+DTE}} b(u_{i,n}, U^*_{n-1}) u^*_{i,n}$

$U_n = \dfrac{U_n}{|b(U_n, U^*_n)|^{1/2}}$ {Normalization}

$U^*_n = \dfrac{U^*_n}{|b(U_n, U^*_n)|^{1/2}}$ {Normalization}

$\Lambda_n = \dfrac{a(U_n, U^*_n)}{b(U_n, U^*_n)}$

The first method is based on Picard's method. The only three parameters not yet defined are M which is the maximum number of mesh adaptation requested, $0 < \text{FIE} \leq \text{TE} \leq \text{FIE} + \text{DTE}$ which is the index of the eigenvalue that the user want to target and err which is tolerance for the a-posteriori error estimator.

Similarly we define the adaptive method based on Newton's method.

Algorithm 7 Adaptive method based on Picard's method

$(\Lambda_M, U_M, U_M^*) = \text{PicardAdapt}(\mathcal{T}_0, \mathcal{T}_0^*, V_0, V_0^*, M, \text{err}, \text{Tol}, \text{AbsTol}, \text{MaxIter}, \text{DTE}, \text{FIE}, \text{TE})$
Construct $\mathcal{T}_0^U$ and V_0^U
Construct $\mathbf{A}_0$, $\mathbf{A}_0^*$, $\mathbf{A}_0^U$ and $\mathbf{B}_0$, $\mathbf{B}_0^*$, $\mathbf{B}_0^U$
$\{(\lambda_{j,0}, u_{j,0}, u_{j,0}^*)\}_{j=1}^{\text{DTE+FIE}} = \text{Eigensolver}(\mathbf{A}_0, \mathbf{A}_0^*, \mathbf{B}_0, \text{DTE} + \text{FIE},$
Tol, MaxIter)
$(\Lambda_0, U_0, U_0^*) = (\lambda_{\text{TE},0}, u_{\text{TE},0}, u_{\text{TE},0}^*)$
$m = 1$
repeat
Construct the mesh $\mathcal{T}_m$ and the finite element space V_m adapting $\mathcal{T}_{m-1}$ and V_{m-1}
Construct the mesh $\mathcal{T}_m^*$ and the finite element space V_m^* adapting $\mathcal{T}_{m-1}^*$ and V_{m-1}^*
Construct $\mathbf{A}_m$, $\mathbf{A}_m^*$, $\mathbf{A}_m^U$ and $\mathbf{B}_m$, $\mathbf{B}_m^*$, $\mathbf{B}_m^U$
$(\lambda_{1,m}, u_{1,m}, u_{1,m}^*) = \text{Picard}(\mathbf{A}_m, \mathbf{A}_m^*, \mathbf{A}_m^U, \mathbf{B}_m, \mathbf{B}_m^*, \mathbf{B}_m^U, u_{1,m-1}, u_{1,m-1}^*, \text{Tol}, \text{AbsTol})$
$j = 1$
for $j = 2$ to $\text{DTE} + \text{FIE}$ **do**
$(\lambda_{j,m}, u_{j,m}, u_{j,m}^*) = \text{PicardOrtho}(\mathbf{A}_m, \mathbf{A}_m^*, \mathbf{A}_m^U, \mathbf{B}_m, \mathbf{B}_m^*, \mathbf{B}_m^U, u_{j,m-1}, u_{j,m-1}^*, \text{Tol},$
$\text{AbsTol}, u_{1,m}, \ldots, u_{j-1,m}, u_{1,m}^*, \ldots, u_{j-1,m}^*)$
end for
$(\Lambda_m, U_m, U_m^*) = \text{Reconstruction}(\{(\lambda_{j,m}, u_{j,m}, u_{j,m}^*)\}_{j=\text{FIE}}^{\text{FIE+DTE}}, (\Lambda_{m-1}, U_{m-1}))$
Compute the a-posteriori error estimator e_m^c as in Algorithm 5
Compute the a-posteriori error estimator $e_m^{c,*}$ as in Algorithm 5
$m = m + 1$
until $m > M$ **or** $\max\{e_{m-1}^c, e_{m-1}^{c,*}\} \leq \text{err}$
$\Lambda_M = \Lambda_{m-1}$
$U_M = U_{m-1}$
$U_M^* = U_{m-1}^*$

10 Iterative methods with improved orthogonalization

The algorithms presented in Sections 5 and 6 are quite costly. When they are used in Algorithm 7 and 8, they ensure that the eigenpair with the correct index TE is computed, but all eigenpairs of indexes from 1 to DTE + FIE need to be computed. Therefore, in this Section we present less computationally expensive methods. The key idea is to employ the orthogonalization only when it is really necessary. This is possible because we can use information from the previous mesh to identify unwanted eigenpairs.

The reason why we introduced the algorithms in Sections 5 and 6 was to cure the downside of the iterative methods to possibly converge to an eigenpair different from the target one. The answer to this problem presented in Sections 5 and 6 was to compute all possible eigenpairs to which the method could erroneously converge to, and then use all of them to force the method, by orthogonalization, to produce an approximation of the wanted eigenpair.

There is a better way which consists in starting without orthogonalization and then every time that the iterative method produces an unwanted eigenpair, save it to be used next time in the orthogonalization process to prevent the method to converge again to the same unwanted solution. This is possible only if a way to distinguish between wanted and unwanted solution is available. In the adaptive setting this is always possible because the orthogonality of any

Algorithm 8 Adaptive method based on Newton's method

$(\Lambda_M, U_M, U_M^*) = \text{NewtonAdapt}(\mathcal{T}_0, \mathcal{T}_0^*, V_0, V_0^*, M, \text{err}, \text{Tol}, \text{AbsTol}, \text{MaxIter}, \text{DTE}, \text{FIE}, \text{TE})$
Construct $\mathcal{T}_0^U$ and V_0^U
Construct $\mathbf{A}_0$, $\mathbf{A}_0^*$, $\mathbf{A}_0^U$ and $\mathbf{B}_0$, $\mathbf{B}_0^*$, $\mathbf{B}_0^U$
$\{(\lambda_{j,0}, u_{j,0}, u_{j,0}^*)\}_{j=1}^{\text{DTE+FIE}} = \text{Eigensolver}(\mathbf{A}_0, \mathbf{A}_0^*, \mathbf{B}_0, \text{DTE} + \text{FIE},$
Tol, MaxIter)
$(\Lambda_0, U_0, U_0^*) = (\lambda_{\text{TE},0}, u_{\text{TE},0}, u_{\text{TE},0}^*)$
$m = 1$
repeat
Construct the mesh $\mathcal{T}_m$ and the finite element space V_m adapting $\mathcal{T}_{m-1}$ and V_{m-1}
Construct the mesh $\mathcal{T}_m^*$ and the finite element space V_m^* adapting $\mathcal{T}_{m-1}^*$ and V_{m-1}^*
Construct $\mathbf{A}_m$, $\mathbf{A}_m^*$, $\mathbf{A}_m^U$ and $\mathbf{B}_m$, $\mathbf{B}_m^*$, $\mathbf{B}_m^U$
$(\lambda_{1,m}, u_{1,m}, u_{1,m}^*) = \text{Newton}(\mathbf{A}_m, \mathbf{A}_m^*, \mathbf{A}_m^U, \mathbf{B}_m, \mathbf{B}_m^*, \mathbf{B}_m^U, u_{1,m-1}, u_{1,m-1}^*, \text{Tol}, \text{AbsTol})$
$j = 1$
for $j = 2$ to $\text{DTE} + \text{FIE}$ **do**
$(\lambda_{j,m}, u_{j,m}, u_{j,m}^*) = \text{NewtonOrtho}(\mathbf{A}_m, \mathbf{A}_m^*, \mathbf{A}_m^U, \mathbf{B}_m, \mathbf{B}_m^*, \mathbf{B}_m^U, u_{j,m-1}, u_{j,m-1}^*, \text{Tol},$
$\text{AbsTol}, u_{1,m}, \ldots, u_{j-1,m}, u_{1,m}^*, \ldots, u_{j-1,m}^*)$
end for
$(\Lambda_m, U_m, U_m^*) = \text{Reconstruction}(\{(\lambda_{j,m}, u_{j,m}, u_{j,m}^*)\}_{j=\text{FIE}}^{\text{FIE+DTE}}, (\Lambda_{m-1}, U_{m-1}))$
Compute the a-posteriori error estimator e_m^c as in Algorithm 5
Compute the a-posteriori error estimator $e_m^{c,*}$ as in Algorithm 5
$m = m + 1$
until $m > M$ **or** $\max\{e_{m-1}^c, e_{m-1}^{c,*}\} \leq \text{err}$
$\Lambda_M = \Lambda_{m-1}$
$U_M = U_{m-1}$
$U_M^* = U_{m-1}^*$

newly computed eigenpair against the results on the previous mesh can be computed.

The Algorithms 9 and 10 are the incarnations of the improved orthogonality technology applied to the either Picard's or Newton's method respectively. Since these two algorithms are identical except for the call to either PicardOrtho or to NewtonOrtho, in the rest we are gong to describe only Algorithm 9.

Algorithm 9 computes a set of eigentriplets $\{(\lambda_{j,n}, u_{j,n}, u_{j,n}^*)\}_{j=\text{FIE}}^{\text{DTE+FIE}}$, approximating the target continuous eigenspaces, on the meshes $\mathcal{T}_n$ and $\mathcal{T}_n^*$. The arguments that it needs are: the matrices $\mathbf{A}$, $\mathbf{A}^*$, $\mathbf{A}^U$, $\mathbf{B}$, $\mathbf{B}^*$, $\mathbf{B}^U$, the approximation of the target triplets $\{(\tilde{\lambda}_{j,n-1}, \tilde{u}_{j,n-1}, \tilde{u}_{j,n-1}^*)\}_{j=\text{FIE}}^{\text{DTE+FIE}}$ computed on the previous meshes $\mathcal{T}_{n-1}$ and $\mathcal{T}_{n-1}^*$, and then projected on the refined meshes $\mathcal{T}_n$ and $\mathcal{T}_n^*$, and a real value $0 < \text{ThO} < 1$ which is used to decide whether a computed eigenfunction is part of the approximation of the target eigenspace or not. The sets $\mathcal{D}$, $\mathcal{D}^*$ are empty at the beginning, but then they are fed with all computed eigenfunctions. Then $\mathcal{D}$ and $\mathcal{D}^*$ are passed to every call to PicardOrtho and so it guarantees that the same eigenfunction is never computed twice. The key part of the algorithm is just after the call to PicardOrtho, where the newly computed eigenfunction is analyzed. The analysis consists in checking how orthogonal the newly computed eigenfunctions $u_{j,n}$, $u_{j,n}^*$ are respect to the span of $\{(\tilde{\lambda}_{j,n-1}, \tilde{u}_{j,n-1}, \tilde{u}_{j,n-1}^*)\}_{j=\text{FIE}}^{\text{DTE+FIE}}$. If the resulting value is smaller than ThO, then $u_{j,n}$, $u_{j,n}^*$ are not considered part

of the target eigenspace and new approximations of $u_{j,n}$, $u^*_{j,n}$ are done. Otherwise, $u_{j,n}$, $u^*_{j,n}$ are kept and the algorithm pass to approximate the next eigenfunctions in the target eigenspace. The algorithm ends when all eigentriplets in $\{(\lambda_{j,n}, u_{j,n}, u^*_{j,n})\}_{j=\mathrm{FIE}}^{\mathrm{DTE+FIE}}$ are computed.

Algorithm 9 Picard's method with improved orthogonalization

$\{(\lambda_{j,n}, u_{j,n}, u^*_{j,n})\}_{j=\mathrm{FIE}}^{\mathrm{DTE+FIE}} = \mathrm{PicardImpOrtho}(\mathbf{A}, \mathbf{A}^*, \mathbf{A}^U, \mathbf{B}, \mathbf{B}^*, \mathbf{B}^U,$
$\quad \{(\tilde{\lambda}_{j,n-1}, \tilde{u}_{j,n-1}, \tilde{u}^*_{j,n-1})\}_{j=\mathrm{FIE}}^{\mathrm{DTE+FIE}}, \mathrm{ThO})$
$\mathcal{D} = \emptyset$
$\mathcal{D}^* = \emptyset$
$j = \mathrm{DTE} + \mathrm{FIE}$
repeat
  if $j \geq \mathrm{FIE}$ **then**
    $(\lambda_{j,n}, u_{j,n}, u^*_{j,n}) = \mathrm{PicardOrtho}(\mathbf{A}, \mathbf{A}^*, \mathbf{A}^U, \mathbf{B}, \mathbf{B}^*, \mathbf{B}^U, \tilde{u}_{j,n-1}, \tilde{u}^*_{j,n-1}, \mathrm{Tol},$
      $\mathrm{AbsTol}, \mathcal{D}, \mathcal{D}^*)$
    Add $u_{j,n}$ to $\mathcal{D}$
    Add $u^*_{j,n}$ to $\mathcal{D}^*$
    $\mathrm{inner} = 0$
    $\mathrm{inner}^* = 0$
    for $i = \mathrm{FIE} \to \mathrm{DTE} + \mathrm{FIE}$ **do**
      $\mathrm{inner} = \mathrm{inner} + (P^* u^*_{j,n})^t \mathbf{B}^U P \tilde{u}_{i,n-1}$
      $\mathrm{inner}^* = \mathrm{inner} + (P^* \tilde{u}^*_{i,n-1})^t \mathbf{B}^U P u_{j,n}$
      $\mathrm{inner} = \max\{\mathrm{inner}, \mathrm{inner}^*\}$
    end for
    if $\mathrm{inner} > \mathrm{ThO}$ **then**
      $j = j - 1$
    end if
  end if
until $j < \mathrm{FIE}$

The number of computed eigenfunction may vary: In the best case scenario when the method is used to approximate an eigenspace of dimension DTE, only DTE eigenfunctions are computed. In the worst case scenario, DTE+FIE eigenfunctions are computed, which is the number of computed eigenfunctions by Algorithms 2, 4 on the same space. Because almost never the worst case scenario is achieved, Algorithms 9 and 10 are more efficient than Algorithms 2, 4. We conclude this section stating the adaptive algorithms with improved orthogonality.

Algorithm 10 Newton's method with improved orthogonalization

$\{(\lambda_{j,n}, u_{j,n}, u^*_{j,n})\}_{j=\mathrm{FIE}}^{\mathrm{DTE+FIE}} = \mathrm{NewtonImpOrtho}(\mathbf{A}, \mathbf{A}^*, \mathbf{A}^U, \mathbf{B}, \mathbf{B}^*, \mathbf{B}^U,$
$\quad \{(\tilde{\lambda}_{j,n-1}, \tilde{u}_{j,n-1}, \tilde{u}^*_{j,n-1})\}_{j=\mathrm{FIE}}^{\mathrm{DTE+FIE}}, \mathrm{ThO})$
$\mathcal{D} = \emptyset$
$\mathcal{D}^* = \emptyset$
$j = \mathrm{DTE} + \mathrm{FIE}$
repeat
  if $j \geq \mathrm{FIE}$ **then**
    $(\lambda_{j,n}, u_{j,n}, u^*_{j,n}) = \mathrm{NewtonOrtho}(\mathbf{A}, \mathbf{A}^*, \mathbf{A}^U, \mathbf{B}, \mathbf{B}^*, \mathbf{B}^U, \tilde{u}_{j,n-1}, \tilde{u}^*_{j,n-1}, \mathrm{Tol},$
      $\mathrm{AbsTol}, \mathcal{D}, \mathcal{D}^*)$
    Add $u_{j,n}$ to $\mathcal{D}$
    Add $u^*_{j,n}$ to $\mathcal{D}^*$
    $\mathrm{inner} = 0$
    $\mathrm{inner}^* = 0$
    for $i = \mathrm{FIE} \to \mathrm{DTE} + \mathrm{FIE}$ **do**
      $\mathrm{inner} = \mathrm{inner} + (P^* u^*_{j,n})^t \mathbf{B}^U P \tilde{u}_{i,n-1}$
      $\mathrm{inner}^* = \mathrm{inner} + (P^* \tilde{u}^*_{i,n-1})^t \mathbf{B}^U P u_{j,n}$
      $\mathrm{inner} = \max\{\mathrm{inner}, \mathrm{inner}^*\}$
    end for
    if $\mathrm{inner} > \mathrm{ThO}$ **then**
      $j = j - 1$
    end if
  end if
until $j < \mathrm{FIE}$

Algorithm 11 Adaptive method based on Picard's method with improved orthogonality

$(\Lambda_M, U_M, U^*_M) = \mathrm{PicardImpAdapt}(\mathcal{T}_0, \mathcal{T}^*_0, V_0, V^*_0, M, \mathrm{err}, \mathrm{Tol}, \mathrm{MaxIter}, \mathrm{DTE}, \mathrm{FIE}, \mathrm{TE}, \mathrm{ThO})$
Construct $\mathbf{A}_0$, $\mathbf{A}^*_0$ and $\mathbf{B}_0$
$\{(\lambda_{j,0}, u_{j,0}, u^*_{j,0})\}_{j=1}^{\mathrm{DTE+FIE}} = \mathrm{Eigensolver}(\mathbf{A}_0, \mathbf{A}^*_0, \mathbf{B}_0, \mathbf{B}^*_0, \mathrm{DTE} + \mathrm{FIE},$
  $\mathrm{Tol}, \mathrm{MaxIter})$
$(\Lambda_0, U_0, U^*_0) = (\lambda_{\mathrm{TE},0}, u_{\mathrm{TE},0}, u^*_{\mathrm{TE},0})$
$m = 1$
repeat
  Construct the mesh $\mathcal{T}_m$ and the finite element space V_m adapting $\mathcal{T}_{m-1}$ and V_{m-1}
  Construct the mesh $\mathcal{T}^*_m$ and the finite element space V^*_m adapting $\mathcal{T}^*_{m-1}$ and V^*_{m-1}
  Construct $\mathbf{A}_m$, $\mathbf{A}^*_m$, $\mathbf{A}^U_m$, $\mathbf{B}_m$, $\mathbf{B}^*_m$, $\mathbf{B}^U_m$
  $\{(\lambda_{j,m}, u_{j,m}, u^*_{j,m})\}_{j=\mathrm{FIE}}^{\mathrm{DTE+FIE}} = \mathrm{PicardImpOrtho}(\mathbf{A}_m, \mathbf{A}^*_m, \mathbf{A}^U_m, \mathbf{B}_m, \mathbf{B}^*_m, \mathbf{B}^U_m,$
    $\{(\tilde{\lambda}_{j,n-1}, \{(\lambda_{j,m-1}, u_{j,m-1}, u^*_{j,m-1})\}_{j=\mathrm{FIE}}^{\mathrm{DTE+FIE}}, \mathrm{ThO})$
  $(\Lambda_m, U_m, U^*_m) = \mathrm{Reconstruction}(\{(\lambda_{j,m}, u_{j,m}, u^*_{j,m})\}_{j=\mathrm{FIE}}^{\mathrm{FIE+DTE}}, (\Lambda_{m-1}, U_{m-1}, U^*_{m-1}))$
  Compute the a-posteriori error estimator e^c_m as in Algorithm 5
  Compute the a-posteriori error estimator $e^{c,*}_m$ as in Algorithm 5
  $m = m + 1$
until $m > M$ **or** $\max\{e^c_{m-1}, e^{c,*}_{m-1}\} \leq \mathrm{err}$
$\Lambda_M = \Lambda_{m-1}$
$U_M = U_{m-1}$
$U^*_M = U^*_{m-1}$

Similarly we define the adaptive method based on Newton's method.

Algorithm 12 Adaptive method based on Newton's method with improved orthogonality

$(\Lambda_M, U_M, U_M^*) = \text{NewtonImpAdapt}(\mathcal{T}_0, \mathcal{T}_0^*, V_0, V_0^*, M, \text{err}, \text{Tol}, \text{MaxIter}, \text{DTE}, \text{FIE}, \text{TE}, \text{ThO})$
Construct $\mathbf{A}_0$, $\mathbf{A}_0^*$ and $\mathbf{B}_0$
$\{(\lambda_{j,0}, u_{j,0}, u_{j,0}^*)\}_{j=1}^{\text{DTE+FIE}} = \text{Eigensolver}(\mathbf{A}_0, \mathbf{A}_0^*, \mathbf{B}_0, \mathbf{B}_0^*, \text{DTE} + \text{FIE},$
$\quad \text{Tol}, \text{MaxIter})$
$(\Lambda_0, U_0, U_0^*) = (\lambda_{\text{TE},0}, u_{\text{TE},0}, u_{\text{TE},0}^*)$
$m = 1$
repeat
  Construct the mesh $\mathcal{T}_m$ and the finite element space V_m adapting $\mathcal{T}_{m-1}$ and V_{m-1}
  Construct the mesh $\mathcal{T}_m^*$ and the finite element space V_m^* adapting $\mathcal{T}_{m-1}^*$ and V_{m-1}^*
  Construct $\mathbf{A}_m$, $\mathbf{A}_m^*$, $\mathbf{A}_m^U$, $\mathbf{B}_m$, $\mathbf{B}_m^*$, $\mathbf{B}_m^U$
  $\{(\lambda_{j,m}, u_{j,m}, u_{j,m}^*)\}_{j=\text{FIE}}^{\text{DTE+FIE}} = \text{NewtonImpOrtho}(\mathbf{A}_m, \mathbf{A}_m^*, \mathbf{A}_m^U, \mathbf{B}_m, \mathbf{B}_m^*, \mathbf{B}_m^U,$
    $\{(\tilde{\lambda}_{j,n-1}, \{(\lambda_{j,m-1}, u_{j,m-1}, u_{j,m-1}^*)\}_{j=\text{FIE}}^{\text{DTE+FIE}}, \text{ThO})$
  $(\Lambda_m, U_m, U_m^*) = \text{Reconstruction}(\{(\lambda_{j,m}, u_{j,m}, u_{j,m}^*)\}_{j=\text{FIE}}^{\text{FIE+DTE}}, (\Lambda_{m-1}, U_{m-1}, U_{m-1}^*))$
  Compute the a-posteriori error estimator e_m^c as in Algorithm 5
  Compute the a-posteriori error estimator $e_m^{c,*}$ as in Algorithm 5
  $m = m + 1$
until $m > M$ **or** $\max\{e_{m-1}^c, e_{m-1}^{c,*}\} \leq \text{err}$
$\Lambda_M = \Lambda_{m-1}$
$U_M = U_{m-1}$
$U_M^* = U_{m-1}^*$

11 A Priori Convergence Results

In this section we present some a priori estimates for non-symmetric eigenvalue problems on independent meshes. To our best knowledge, this is the first time such results are presented.

The results prove exponential convergence of the method under uniform refinement and cast some hopes that the same kind of convergence is achieved with adaptive refinement as well. This will be confirmed by the numerical results in Section 12.3. Moreover, in Theorem 4 and Theorem 5 we prove a priori convergence results for the reconstructed triplet (Λ_n, U_n, U_n^*) for both coercive and non-coercive bilinear forms $a(\cdot, \cdot)$.

The distance of an approximate eigenfunction from the true eigenspace is a crucial quantity in the convergence analysis for eigenvalue problems especially in the case of non-simple eigenvalues.

Definition 1 Given a function $v \in L^2(\Omega)$ and a finite dimensional subspace $\mathcal{P} \subset L^2(\Omega)$, we define:

$$\text{dist}(v, \mathcal{P})_1 \;=\; \min_{w \in \mathcal{P}} \|v - w\|_1.$$

From now on we shall let C denote a generic constant which may depend on the true eigenvalues and vectors of (3) and other constants introduced above, but is always independent of n, as well as h_n, h_n^*, p_n and p_n^*. The next lemma comes from the results in [3] and from standard *hp*-FEM results.

Lemma 1 *Suppose that the bilinear form $a(\cdot,\cdot)$ is coercive and suppose $1 \leq j \leq \min\{\dim V_n, \dim V_n^*\}$. Let λ_j be an eigenvalue with corresponding eigenspaces $E(\lambda_j), E^*(\lambda_j) \subset H^{1+\mu}(\Omega)$, for $\mu > 1$, of any (finite) dimension and let $(\lambda_{j,n}, u_{j,n}u^*_{j,n})$ be an eigentriplet of (6). Then, for finite element spaces V_n and V_n^* sufficiently rich,*

(i)

$$|\lambda_j - \lambda_{j,n}| \;\leq\; C\frac{h_n^\mu}{p_n^\mu}\frac{(h_n^*)^\mu}{(p_n^*)^\mu}; \tag{15}$$

(ii)

$$\mathrm{dist}(u_{j,n}, E_1(\lambda_j))_1 \;\leq C\frac{h_n^\mu}{p_n^\mu}, \tag{16}$$

(iii)

$$\mathrm{dist}(u^*_{j,n}, E_1^*(\lambda_j))_1 \;\leq C\frac{(h_n^*)^\mu}{(p_n^*)^\mu}, \tag{17}$$

with $1 \leq \mu \leq \min\{p_n, p_n^*\}$.

Proof In this proof we use th results in [3], those results are stated for discrete and continuous eigenfunctions normalized in the H^1 norm, However in this work we normalize the eigenfunctions in a different way. It is however easy to see that the change in the normalization does not affect the final results. In fact let $\tilde{u}$, $\tilde{u}^*$ be the left and right eigenfunctions corresponding to the same eigenvalue where both functions are normalized in the H^1 norm and let u_n, u_n^* be the computed approximations using one of the methods presented in this work, then denoting by $u = \tilde{u}/|b(\tilde{u},\tilde{u}^*)|^{1/2}$ and by $u_n = \tilde{u}_n/|b(\tilde{u}_n,\tilde{u}_n^*)|^{1/2}$, where $\tilde{u}_n$, $\tilde{u}_n^*$ are u_n, u_n^* normalized in H^1 and supposing that both $\tilde{u} - \tilde{u}_n$ and $\tilde{u}^* - \tilde{u}_n^*$ are converging, we have that

$$u - u_n = \frac{\tilde{u}}{|b(\tilde{u},\tilde{u}^*)|^{1/2}} - \frac{\tilde{u}_n}{|b(\tilde{u}_n,\tilde{u}_n^*)|^{1/2}}$$

$$= |b(\tilde{u},\tilde{u}^*)|^{-1/2}(\tilde{u} - \tilde{u}_n) + u_n(|b(\tilde{u},\tilde{u}^*)|^{-1/2} - |b(\tilde{u}_n,\tilde{u}_n^*)|^{-1/2}) \;,$$

and

$$u^* - u_n^* = \frac{\tilde{u}^*}{|b(\tilde{u},\tilde{u}^*)|^{1/2}} - \frac{\tilde{u}_n^*}{|b(\tilde{u}_n,\tilde{u}_n^*)|^{1/2}}$$

$$= |b(\tilde{u},\tilde{u}^*)|^{-1/2}(\tilde{u}^* - \tilde{u}_n^*) + u_n^*(|b(\tilde{u},\tilde{u}^*)|^{-1/2} - |b(\tilde{u}_n,\tilde{u}_n^*)|^{-1/2}) \;,$$

where the second sequences on the rhs converges by the assumption that the sequences normalized in H^1 converges.

First consider part (i), the estimate in (15) comes from Theorem 8.3 in [3] which gives

$$|\lambda_j - \lambda_{j,n}| \;\leq\; C \sup_{u\in E_1(\lambda_j)} \inf_{v_n\in V_n} \|u - v_n\|_1 \sup_{u^*\in E_1^*(\lambda_j)} \inf_{v_n^*\in V_n^*} \|u^* - v_n^*\|_1.$$

Combining this with standard finite element error estimates for *hp*-method, we get

$$|\lambda_{j,n} - \lambda_j| \quad \leq \; C \frac{h_n^{\min(\mu,p)}}{p_n^{\mu}} \sup_{u \in E_1(\lambda_j)} \|u\|_{1+\mu} \frac{(h_n^*)^{\min(\mu,p^*)}}{(p_n^*)^{\mu}} \sup_{u^* \in E_1^*(\lambda_j)} \|u^*\|_{1+\mu}, \tag{18}$$

To obtain (ii), we use Theorem 8.4 in [3]:

$$\mathrm{dist}(u_{j,n}, E_1(\lambda_j))_1 \; \leq \; C \sup_{u \in E_1(\lambda_j)} \inf_{v_n \in V_n} \|u - v_n\|_1 \; \leq C \frac{h_n^{\min(\mu,p)}}{p_n^{\mu}} \sup_{u \in E_1(\lambda_j)} \|u\|_{1+\mu} \, .$$

The proof of (iii) is analogous to (ii).

The next theorem shows that also the reconstructed triplet (Λ_n, U_n, U_n^*) converges in a similar way to standard computed eigenpairs. It is interesting to remind that in general (Λ_n, U_n, U_n^*) is not an eigentriplet of the discrete problem (6).

Theorem 4 *Suppose that the bilinear form $a(\cdot,\cdot)$ is coercive and suppose $1 \leq j \leq \min\{\dim V_n, \dim V_n^*\}$. Let λ_j be an eigenvalue with corresponding eigenspaces $E(\lambda_j), E^*(\lambda_j) \subset H^{1+\mu}(\Omega)$, for $\mu > 1$, of any (finite) dimension and let (Λ_n, U_n, U_n^*) be an reconstructed triple of* (6)*. Then, for finite element spaces V_n and V_n^* sufficiently rich,*

(i)

$$|\lambda_j - \Lambda_n| \; \leq \; C \frac{h_n^{\mu}}{p_n^{\mu}} \frac{(h_n^*)^{\mu}}{(p_n^*)^{\mu}}; \tag{19}$$

(ii)

$$\mathrm{dist}(U_n, E_1(\lambda_j))_1 \; \leq C \frac{h_n^{\mu}}{p_n^{\mu}}, \tag{20}$$

(iii)

$$\mathrm{dist}(U_n^*, E_1^*(\lambda_j))_1 \; \leq C \frac{(h_n^*)^{\mu}}{(p_n^*)^{\mu}}, \tag{21}$$

with $1 \leq \mu \leq \min\{p_n, p_n^*\}$.

Proof Denoting by

$$U = \sum_{i=1}^{R} c_i \; u_i, \quad U^* = \sum_{i=1}^{R} c_i^* \; u_i^*,$$

we have that

$$\mathrm{dist}(U_n, E_1(\lambda_j))_1 \leq \|U - U_n\|_1 \leq \sum_{i=1}^{R} \mathrm{dist}(u_{i,n}, E_1(\lambda_j))_1.$$

and

$$\mathrm{dist}(U_n^*, E_1^*(\lambda_j))_1 \leq \|U^* - U_n^*\|_1 \leq \sum_{i=1}^{R} \mathrm{dist}(u_{i,n}^*, E_1^*(\lambda_j))_1.$$

Then results (ii) and (iii) comes straightforwardly form Lemma 1(ii-iii).

From the canonical form result for compact operators, see [22, Theorem 9.17] we have for any u_i, u_j^* eigenfunctions with $i \neq j$ that $a(u_i, u_j^*) = b(u_i, u_j^*) = 0$. Similarly by construction also the computed eigenfunctions satisfy $a(u_{i,n}, u_{j,n}^*) = b(u_{i,n}, u_{j,n}^*) = 0$ for any $i \neq j$. So from the reconstruction process we have that

$$\Lambda_n = \frac{a(U_n, U_n^*)}{b(U_n, U_n^*)} = \frac{\sum_{i=1}^R c_i c_i^* a(u_{i,n}, u_{i,n}^*)}{\sum_{i=1}^R c_i c_i^*} = \frac{\sum_{i=1}^R c_i c_i^* \lambda_{i,n}}{\sum_{i=1}^R c_i c_i^*} \ .$$

In a similar way we have from the definition of the continuous problems:

$$\frac{a(U, U^*)}{b(U, U^*)} = \frac{\sum_{i=1}^R c_i c_i^* a(u_i, u_i^*)}{\sum_{i=1}^R c_i c_i^*} = \lambda \ .$$

Then

$$|\lambda - \Lambda_n| = \sum_{i=1}^R \frac{|c_i c_i^*|}{|\sum_{i=1}^R c_i c_i^*|} |\lambda - \lambda_{i,n}| \ ,$$

Then the result comes from Lemma 1(i).

The bilinear form $a(\cdot, \cdot)$ is not necessary coercive, but from Gårding's inequality, see [23, Theorem (5.6.8)], we have that either $a(\cdot, \cdot)$ or $a(\cdot, \cdot)$ with a shift is coercive, i.e., $a(\cdot, \cdot) + Kb(\cdot, \cdot)$, with $K > 0$. In the latter case, we can define a new bilinear form $\tilde{a}(\cdot, \cdot) = a(\cdot, \cdot) + Kb(\cdot, \cdot)$ and all the results in this section still hold. Moreover it is easy to see that λ is an eigenvalue for $a(\cdot, \cdot)$ if and only if $\sigma = \lambda + K$ is an eigenvalue of $\tilde{a}(\cdot, \cdot)$. Also (λ, u) and (λ, u^*) are left and right eigenpairs of $a(\cdot, \cdot)$ if and only if (σ, u) and (σ, u^*) are left and right eigenpairs of $\tilde{a}(\cdot, \cdot)$. This is also true for the discrete spectra, so the action of the shift $Kb(\cdot, \cdot)$ shifts the spectrum by K, but keeps the eigenfunctions. In particular it is not necessary to use the shifted form $\tilde{a}(\cdot, \cdot)$ in practice, because all methods presented in this paper work also if $a(\cdot, \cdot)$ is not coercive and the equivalence of the discrete spectra does the rest. Now we recast the previous two results for a not coercive $a(\cdot, \cdot)$. The proofs are straightforward because the results in [3] hold also for $\tilde{a}(\cdot, \cdot)$.

Lemma 2 *Suppose that the bilinear form $a(\cdot, \cdot)$ is not coercive, but $\tilde{a}(\cdot, \cdot)$ is coercive. Suppose $1 \leq j \leq \min\{\dim V_n, \dim V_n^*\}$. Let λ_j be an eigenvalue with corresponding eigenspaces $E(\lambda_j), E^*(\lambda_j) \subset H^{1+\mu}(\Omega)$, for $\mu > 1$, of any (finite) dimension and let $(\lambda_{j,n}, u_{j,n} u_{j,n}^*)$ be an eigentriplet of (6) and $(\sigma_{j,n}, u_{j,n}, u_{j,n}^*)$ is the corresponding discrete eigentriplet for $\tilde{a}(\cdot, \cdot)$. Then, for finite element spaces V_n and V_n^* sufficiently rich,*

(i)

$$|\lambda_j - \lambda_{j,n}| \ = \ |\lambda_j + K - \lambda_{j,n} - K| \ = \ |\sigma_j - \sigma_{j,n}| \ \leq \ C \frac{h_n^\mu}{p_n^\mu} \frac{(h_n^*)^\mu}{(p_n^*)^\mu}, \quad (22)$$

(ii)

$$\text{dist}(u_{j,n}, E_1(\lambda_j))_1 \ \leq C\frac{h_n^\mu}{p_n^\mu}, \tag{23}$$

(iii)

$$\text{dist}(u^*_{j,n}, E^*_1(\lambda_j))_1 \ \leq C\frac{(h^*_n)^\mu}{(p^*_n)^\mu}, \tag{24}$$

with $1 \leq \mu \leq \min\{p_n, p^*_n\}$.

The proofs of (ii) and (iii) are trivial because the eigenfunctions are not modified by the shift in $\tilde{a}(\cdot,\cdot)$.

Theorem 5 *Suppose that the bilinear form $a(\cdot,\cdot)$ is coercive and suppose $1 \leq j \leq \min\{\dim V_n, \dim V^*_n\}$. Let λ_j be an eigenvalue with corresponding eigenspaces $E(\lambda_j), E^*(\lambda_j) \subset H^{1+\mu}(\Omega)$, for $\mu > 1$, of any (finite) dimension and let (Λ_n, U_n, U^*_n) be the reconstructed triplet of (6) and (Σ_n, U_n, U^*_n) is the corresponding discrete triplet for $\tilde{a}(\cdot,\cdot)$. Then, for finite element spaces V_n and V^*_n sufficiently rich,*

(i)

$$|\sigma_j - \Sigma_n| \ = \ |\lambda_j - \Lambda_n| \ \leq \ C\frac{h_n^\mu}{p_n^\mu}\frac{(h^*_n)^\mu}{(p^*_n)^\mu}; \tag{25}$$

(ii)

$$\text{dist}(U_n, E_1(\lambda_j))_1 \ \leq C\frac{h_n^\mu}{p_n^\mu}, \tag{26}$$

(iii)

$$\text{dist}(U^*_n, E^*_1(\lambda_j))_1 \ \leq C\frac{(h^*_n)^\mu}{(p^*_n)^\mu}, \tag{27}$$

with $1 \leq \mu \leq \min\{p_n, p^*_n\}$.

12 Numerical Results

12.1 Orthogonality technologies

In this first set of examples, we would like to show the advantages of the orthogonality technologies presented in Sections 5, 6 and 10. We want to approximate the fifth eigenvalue and the corresponding eigenfunction of problem (1) on the square domain $[0,\pi]^2$ and with $b = (0.5, 0.5)$ and $c = 0$ just calling Picard's method as in Algorithm 1 with no orthogonalization and starting with four quadratic elements forming the initial structured mesh. So, as always, we compute the approximation of the first fifth eigenpair on the initial coarse mesh with a generalized eigensolver, then we adapt the mesh for the fifth eigenpair and from that point on we use Picard's method. As can be seen from the piece of output below, already on the first adapted mesh Picard's method goes away from the correct value 98.8210440108936 for the eigenvalue and converges closer to the first eigenvalue 19.8642088021787:

```
- Picard iter 1,  ndof 121, eigenvalue: 100.63886872115, err_rel 82.1409%
- Picard iter 2,  ndof 121, eigenvalue: 72.061449013246, err_rel 26.0106%
- Picard iter 3,  ndof 121, eigenvalue: 24.048653326668, err_rel 189.064%
- Picard iter 4,  ndof 121, eigenvalue: 19.896859692653, err_rel 119.932%
- Picard iter 5,  ndof 121, eigenvalue: 19.842805664953, err_rel 26.1302%
- Picard iter 6,  ndof 121, eigenvalue: 19.859718273742, err_rel 5.49526%
- Picard iter 7,  ndof 121, eigenvalue: 19.863473026642, err_rel 1.30883%
- Picard iter 8,  ndof 121, eigenvalue: 19.864119690489, err_rel 0.395458%
- Picard iter 9,  ndof 121, eigenvalue: 19.864225593476, err_rel 0.144903%
- Picard iter 10, ndof 121, eigenvalue: 19.864242730568, err_rel 0.0571376%
- Picard iter 11, ndof 121, eigenvalue: 19.864245495476, err_rel 0.0229224%
- Picard iter 12, ndof 121, eigenvalue: 19.864245941239, err_rel 0.0092188%
- Picard iter 13, ndof 121, eigenvalue: 19.864246013093, err_rel 0.00370628%
- Picard iter 14, ndof 121, eigenvalue: 19.864246024674, err_rel 0.00148921%
- Picard iter 15, ndof 121, eigenvalue: 19.864246026541, err_rel 0.000598148%
- Picard iter 16, ndof 121, eigenvalue: 19.864246026842, err_rel 0.000240197%
- Picard iter 17, ndof 121, eigenvalue: 19.864246026890, err_rel 9.64444e-05%
```

This is a clear example of what was predicted in Theorem 1. On the other hand, using the standard orthogonality (Section 5) we cure this problem and the method converges to the correct eigenpair.

12.2 Approximating eigenfunctions on individual meshes

Next we would like to illustrate that in general each eigenfunction should be approximated on its own mesh. We choose the L-shape domain with $b = (0.5, 0.5)$ and $c = 0$ where the eigenfunctions of the first eigenvalue exhibit singularities in the gradient at the re-entrant corner while the eigenfunctions of the second eigenvalue are completely smooth. The differences in the regularity are reflected in the adapted meshes: For the first eigenfunction a great amount of h-refinement takes place at the re-entrant corner. For the second eigenfunction we have an adapted mesh mostly characterized by p-refinement. Moreover the fact that the operator in non-symmetric affects in different ways the left and right eigenfunctions. In fact as can be seen the strength of the singularity at the re-entrant corner is different in the left and right eigenfunctions for the first eigenvalue and this is reflected in differences in the corresponding adapted meshes. Similarly also the left and right eigenfunctions of the second eigenvalue are not the same, even if both are smooth in this case, and this is again reflected in differences in the corresponding adapted meshes. These results are shown in Figs. 3, 4, 5 and 6.

12.3 Domains with few reentering corners

We conclude the numerics section considering the model problem (1) on the square domain Ω with a square hole and with $b = (1, 0)$ and $c = 0$. Assuming that we are interested in the eigenfunctions corresponding to the first eigenvalue, see Fig. 7 and that we use the initial mesh in Fig. 8, we want to study

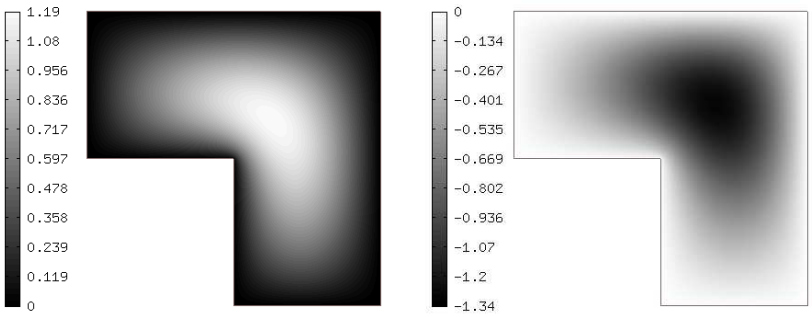

Fig. 3 First eigenfunctions for the first eigenvalue of the L-shaped domain.

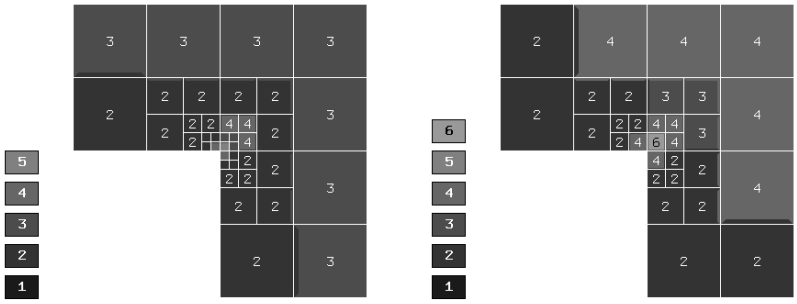

Fig. 4 Adapted meshes with polynomial degrees after 8 adaptive steps for the eigenfunctions in Fig. 3.

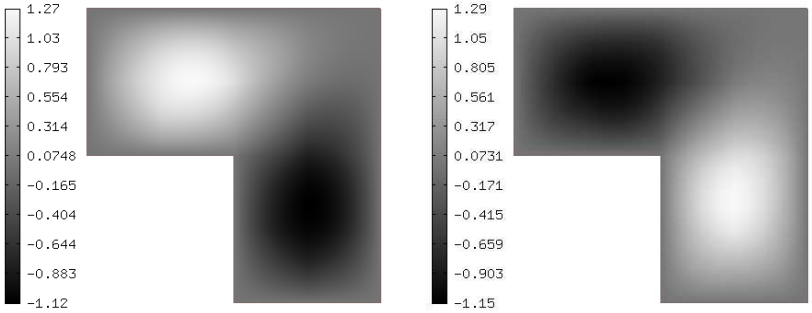

Fig. 5 First eigenfunctions for the first eigenvalue of the L-shaped domain.

the convergence of our *hp*-adapted method on independent meshes based on Picard's method, i.e., Algorithm 11.

In Fig. 9 we present the adapted meshes for the left and right eigenfunctions after 12 applications of the adaptive procedure. As can be seen the two

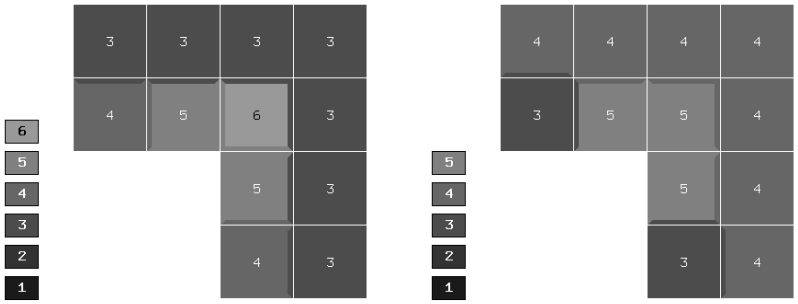

Fig. 6 Adapted meshes with polynomial degrees after 8 adaptive steps for the eigenfunctions in Fig. 5.

meshes are different in order to fit with the different characteristics of each eigenfunction.

In Fig. 10 the number of degrees of freedoms are plotted against the errors from both the left and right eigenfunctions. As can be seen, even if the eigenfunctions are different, both meshes are adapted in such a way that the convergence rate is very similar. Moreover the fact that the curves in Fig. 10 seem to approximate straight lines, which suggests exponential convergence rate. In Fig. 11 the errors are plotted against the number of adapted steps. Also in this case the convergence look exponential for both eigenfunctions. Finally in Fig. 12 we report the number of iterations at each adaptive step necessary to Picard's method in Algorithm 11 to converge with relative error $1e-3$ and absolute error $1e-9$. On average the number of iterations seems to reduce after each adaptation of the meshes and after the 14th adaptation the number of iterations settle to 2. This is particularly interesting from a numerical point of view because the number of iterations are minimum where the linear system is bigger making the method quite cheap to use.

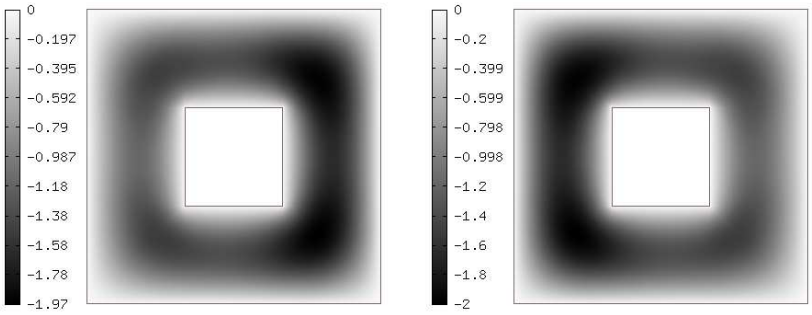

Fig. 7 Left and right eigenfunctions corresponding to the first eigenvalue.

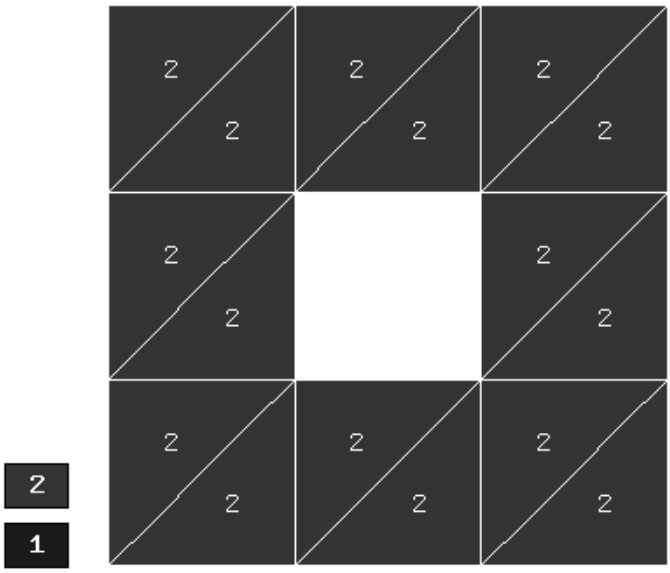

Fig. 8 Initial mesh with polynomial degrees.

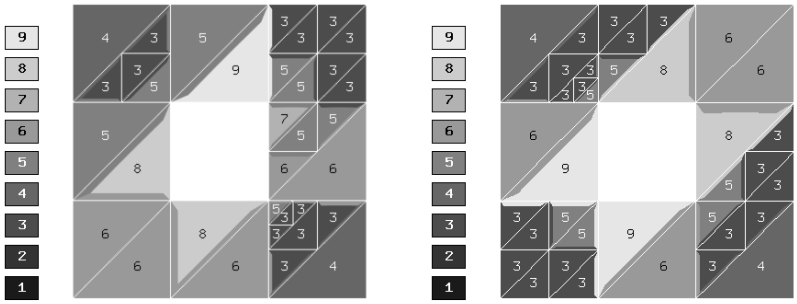

Fig. 9 *hp*-adapted meshes with polynomial degrees for the left and right eigenfunctions after 12 adaptations.

13 Conclusion and Outlook

In this paper we have generalized the results of [19] to non-symmetric eigenvalue problems. Through many numerical experiments, of which just a few were presented here, we gained confidence that the methods work well in practice.

The major contribution of these results is that one does not have to call the generalized eigensolver in every adaptivity step and always compute all eigenpairs or eigentriplets. Instead, one can select one or a few of them and

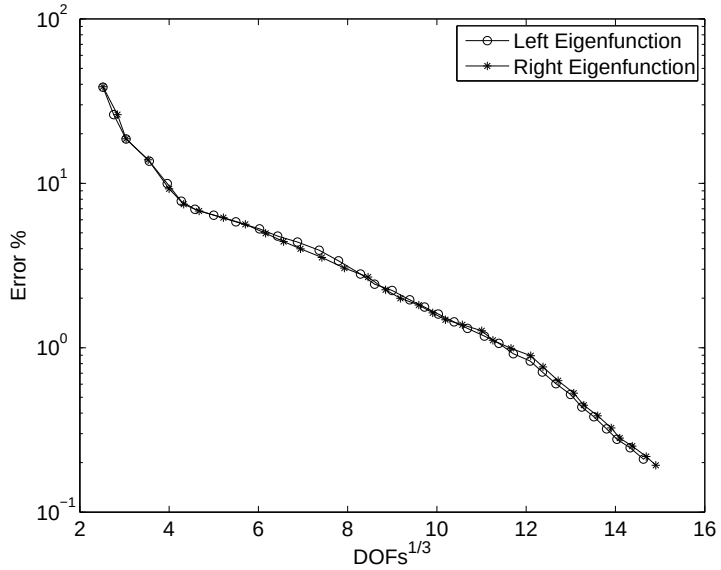

Fig. 10 Convergence plots in number of degrees of freedom.

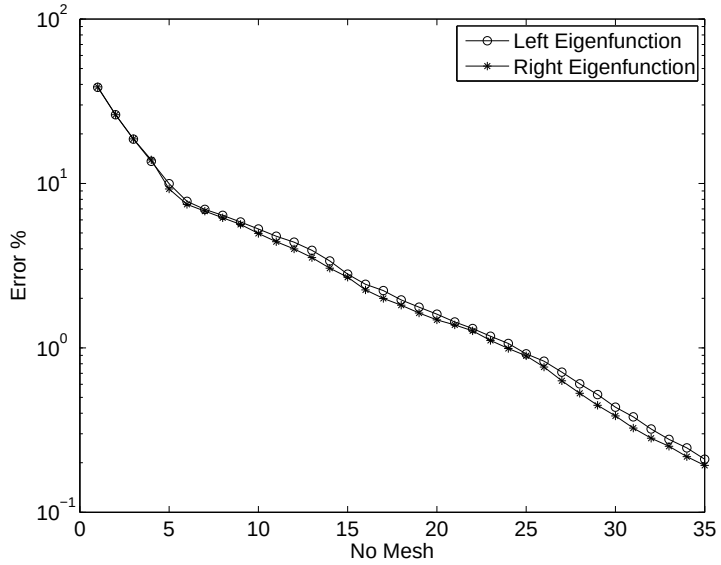

Fig. 11 Convergence plots in number of mesh adaptations.

resolve them adaptively on meshes that moreover evolve differently during the adaptivity process.

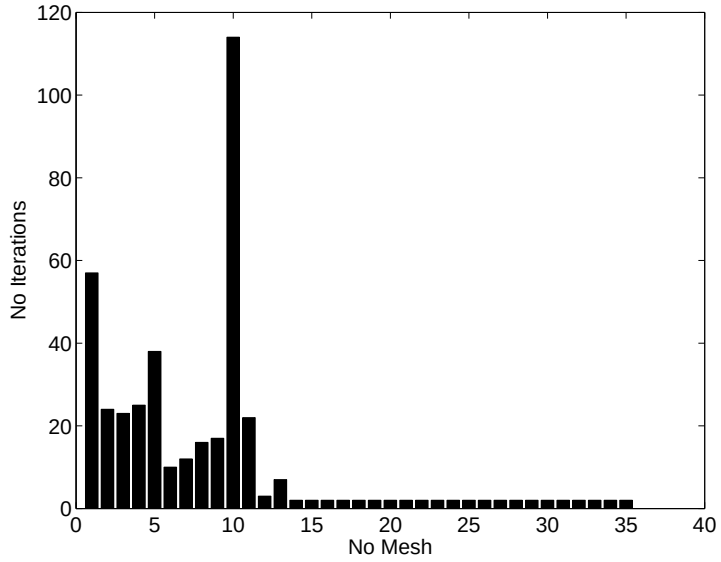

Fig. 12 Number of iterations for Picard's method after each mesh adaptation.

In the next steps, we will employ the multi-mesh *hp*-FEM to actually do this. Our target applications will be neutronics and the Density Functional Theory (DFT) in quantum chemistry.

Acknowledgments

The first author was supported by the Subcontract No. 00089911 of Battelle Energy Alliance (U.S. Department of Energy intermediary) as well as by the Grant No. P105/10/1682 of the Grant Agency of the Czech Republic.

References

1. D. G. Anderson, Iterative procedures for nonlinear integral equations, J. Assoc. Comput. Machinery, 12 (1965) 547–560.
2. I. Babuška, J. Osborn, Estimates for the errors in eigenvalue and eigenvector approximation by Galerkin methods, with particular attention to the case of multiple eigenvalues, SIAM J. Numer. Anal., 24 (1987) 1249–1276.
3. I. Babuška, J. Osborn, Eigenvalue problems, in Handbook of Numerical Analysis Vol II, eds P.G. Cairlet and J.L. Lions, North Holland, 1991.
4. I. Babuška, J. Osborn, Finite element-Galerkin approximation of the eigenvalues and eigenvectors of selfadjoint problems, Math. Comput. 186 (1989) 275–297.
5. P. Berini, Plasmon-polariton waves guided by thin lossy metal films of finite width: Bound modes of symmetric structures, Physical Review B 61:15 (2000) 10484+.
6. A. Cliffe, E. Hall, P. Houston, Adaptive discontinuous Galerkin methods for eigenvalue problems arising in incompressible fluid flows, SIAM Journal on Scientific Computing, 31:6 (2010) 4607–4632.

7. L. Dubcova, P. Solin, G. Hansen, H. Park, Comparison of multimesh *hp*-fem to interpolation and projection methods for spatial coupling of reactor thermal and neutron diffusion calculations, J. Comput. Phys. 230 (2011) 1182–1197.
8. S. Giani, Convergence of Adaptive Finite Element Methods for Elliptic Eigenvalue Problems with Application to Photonic Crystals, Ph.D. thesis, Department of Mathematical Sciences, University of Bath, 2008.
9. S. Giani, I. G. Graham, A convergent adaptive method for elliptic eigenvalue problems, SIAM J. Numer. Anal. 47:2 (2009) 1067–1091.
10. S. Giani, I. G. Graham, Adaptive finite element methods for computing band gaps in photonic crystals, Numerische Mathematik, accepted.
11. L. Grubišić, J. S. Ovall, On estimators for eigenvalue/eigenvector approximations, Mathematics of Computation, 78:266 (2008) 739–770.
12. W. Hackbusch, Elliptic Differential Equations, Springer, Berlin, 1992.
13. J. D. Joannopoulos, R. D. Meade, J. N. Winn, Photonic crystals. Molding the flow of light Princeton Univ. Press, Princeton, NJ, 1995.
14. N. Lalor, H-H. Priebsch, The prediction of low- and mid-frequency internal road vehicle noise: a literature survey, in Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering 221:3 (2007) 245–269.
15. R. B. Lehoucq, D. C. Sorensen, C. Yang, ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted arnoldi methods, SIAM, 1998.
16. H. Röck, Finfding an eigenvector and eigenvalue, with Newtons method for solving systems of nonlinear equations, Report CMDIE WS2002/03
17. P. Solin, D. Andrs, J. Cerveny, M. Simko, PDE-independent adaptive *hp*-fem based on hierarchic extension of finite element spaces, J. Comput. Appl. Math., 233 (2010), 3086–3094.
18. P. Solin, J. Cerveny, L. Dubcova, D. Andrs, Monolithic discretization of linear thermoelasticity problems via adaptive multimesh *hp*-FEM, J. Comput. Appl. Math, 234 (2010) 2350–2357.
19. P. Solin, S. Giani, An iterative finite element method for elliptic eigenvalue problems, Journal of Computational and Applied Mathematics, Vol. 236, No. 18., pp. 4582-4599.
20. P. Solin, K. Segeth, I. Dolezel, Higher-order finite element methods, Chapman & Hall, CRC Press, London, 2003.
21. G. Strang, G. J. Fix, An analysis of the finite element method, Prentice-Hall, 1973.
22. P. D. Hislop, I. M. Sigal, Introduction to spectral theory, Springer, 1996.
23. S. C. Brenner, L. R. Scott, The mathematical theory of finite element methods, Springer, 1994.